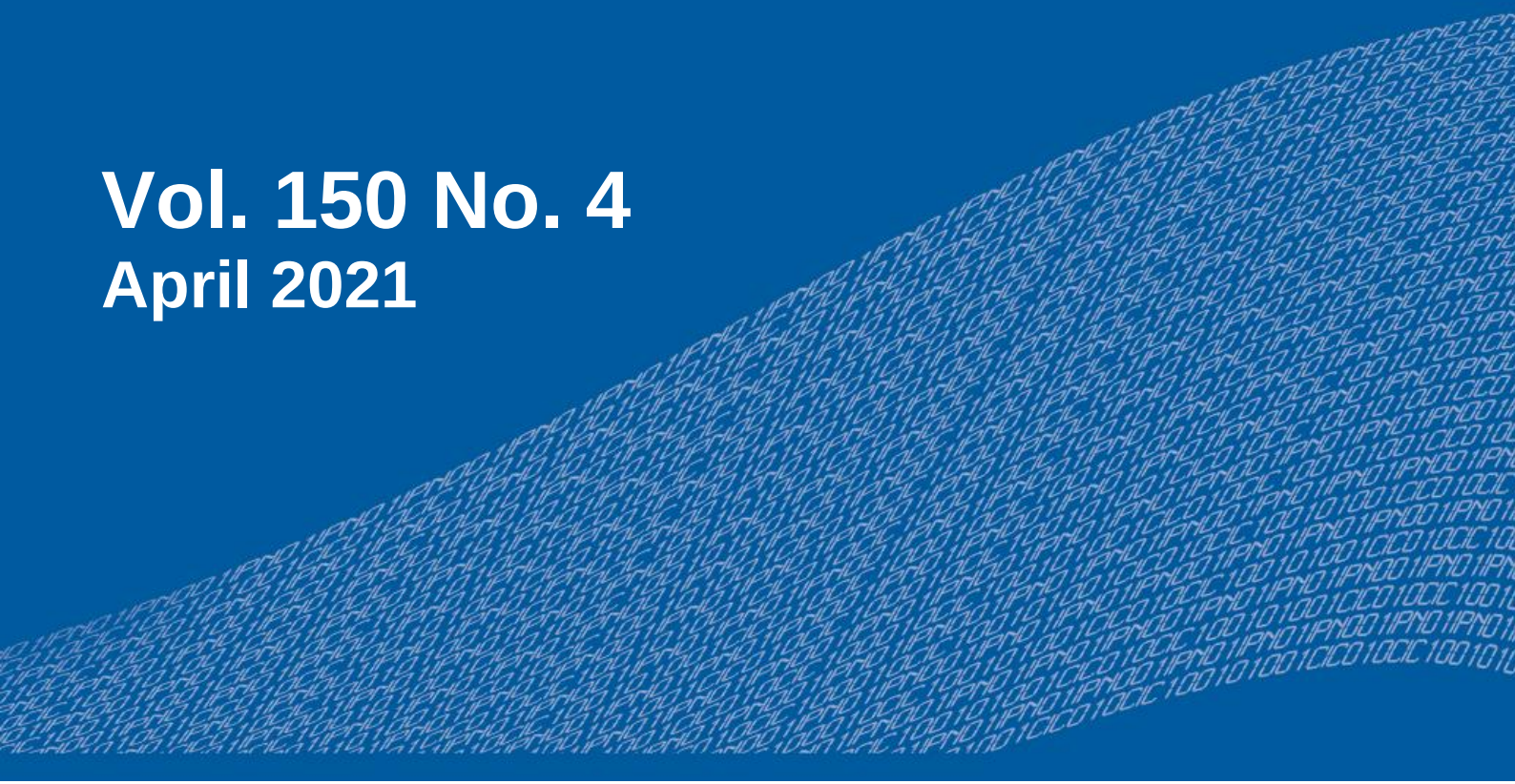
EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

Instituto Politécnico Nacional
"La Técnica al Servicio de la Patria"

Research in Computing Science

ISSN: 1870-4069



Vol. 150 No. 4
April 2021

Research in Computing Science

Research in Computing Science, Año 20, Volumen 150, No. 4, abril de 2021, es una publicación mensual, editada por el Instituto Politécnico Nacional, a través del Centro de Investigación en Computación. Av. Juan de Dios Bátiz S/N, Esq. Av. Miguel Othon de Mendizábal, Col. Nueva Industrial Vallejo, C.P. 07738, Ciudad de México, Tel. 57 29 60 00, ext. 56571. https://www.rcs.cic.ipn.mx. Editor responsable: Dr. Grigori Sidorov. Reserva de Derechos al Uso Exclusivo del Título No. 04-2019-082310242100-203. ISSN: en trámite, ambos otorgados por el Instituto Politécnico Nacional de Derecho de Autor. Responsable de la última actualización de este número: el Centro de Investigación en Computación, Dr. Grigori Sidorov, Av. Juan de Dios Bátiz S/N, Esq. Av. Miguel Othon de Mendizábal, Col. Nueva Industrial Vallejo, C.P. 07738. Fecha de última modificación 01 de diciembre de 2020.

Research in Computing Science, year 20, Volume 150, No. 4, April 2021, is published monthly by the Center for Computing Research of IPN.

Volume 150(4)

Advances in Information Technology

Germán Ríos Toledo
Fernando Pech May (eds.)

Instituto Politécnico Nacional, Centro de Investigación en Computación
México 2021

ISSN: in process

Formerly ISSNs: 1870-4069, 1665-9899.

Instituto Politécnico Nacional (IPN)
Centro de Investigación en Computación (CIC)
Av. Juan de Dios Bátiz s/n esq. M. Othón de Mendizábal
Unidad Profesional "Adolfo López Mateos", Zacatenco
07738, México D.F., México

http://www.rcs.cic.ipn.mx
http://www.ipn.mx
http://www.cic.ipn.mx

Indexed in LATINDEX, DBLP and Periodica

Electronic edition

ISSN 1870-4069

Table of Contents

Page

ISSN 1870-4069

Diseño conceptual de un sistema embebido para seguridad en puertas de acceso y ventanas en hogares

Adrián Cristino-Batalla, Kassandra Rubí Nava-Guerrero, Derlis Hernández-Lara, Rafael Pérez-Rojas

Tecnológico Nacional de México,
Tecnológico de Estudios Superiores de Ecatepec,
Estado de México, Mexico

{201721714, 201721738, dderlis-lara, 201621329}@tese.edu.mx

Resumen. Este trabajo presenta el diseño de un sistema de seguridad en puertas de acceso y ventanas de hogares, relacionado con la domótica que es el nombre que se le da a los sistemas que automatizan a una vivienda, generando bienestar, enfocados en seguridad, gestión de servicios de energía eléctrica y agua. Con el fin de brindar una herramienta que otorgue mayor seguridad en viviendas. Para la metodología se utilizó *Design Thinking* ya que esta herramienta permite desarrollar productos basados en las necesidades del usuario final. Por lo que se diseñó un prototipo de sistema embebido que incluye un panel principal manejado por un *keypad* (teclado numérico) que realiza la función del cierre individual de puertas y ventanas, controlado por contraseña, y en caso de ser incorrecta se habilitará un speaker para alertar de un acceso no valido, impidiendo el ingreso al sistema, colocado en los accesos del hogar. Tomando en cuenta que en México la seguridad en casas es deficiente con respecto a las estadísticas obtenidas del INEGI y SESNSP, del 2018 en adelante.

Palabras clave: Domótica, seguridad, *Design Thinking*.

Conceptual Design of an Embedded System for Security in Access Doors and Windows in Homes

Abstract. This work presents the design of a security system in access doors and windows of homes, related to home automation, it is the name given to systems that automate a home, generating well-being, focused on security, service management of electricity and water. In order to provide a tool that provides greater security in homes. Design Thinking was used for the methodology, since this tool allows the development of products based on the needs of the end user. Therefore, a prototype of an embedded system was designed that includes a main panel managed by a keypad (numeric keyboard) that performs the function of individual closing of doors and windows, controlled by password, and in case of being incorrect, a speaker will be enabled to warn of invalid access, preventing entry to the system, placed at home entrances. Taking into account that in Mexico home security is deficient with respect to the statistics obtained from INEGI and SESNSP, from 2018 onwards.

Keywords: Home automation, security, *Design Thinking*.

1. Introducción

La inseguridad en México a nivel nacional ya no solo son asaltos con armas, robo de automóviles o extorsión, de acuerdo con los resultados del vigésimo sexto levantamiento de la Encuesta Nacional de Seguridad Pública Urbana (ENSU), realizada por el INEGI, en la primera quincena de diciembre de 2019, durante ese mes 72.9% de la población de 18 años y más consideró que vivir en su ciudad es inseguro, actualmente se abrieron 637 carpetas de investigación relacionado con robos a casa habitación dejando al estado de México en primer lugar como se muestra en la Figura 1 [1]. Teniendo en cuenta la deficiencia que se tiene por parte de las autoridades para atender llamados por motivos de inseguridad, llegando a tiempos excedentes para acudir al sitio del delito, dentro de las soluciones que se toman para prevenir y tener pruebas se opta por colocar cámaras de seguridad CCTV (Circuito Cerrado de Televisión), alarmas vecinales, cercas eléctricas o alambrado de púas.

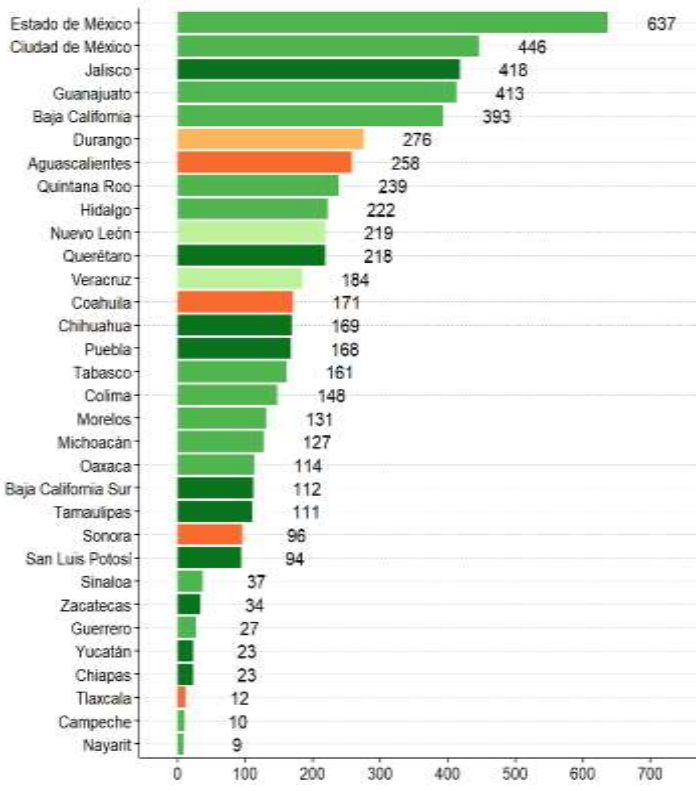


Fig. 1. Carpetas de investigación abiertas de registro de robos a casa habitación de mayor a menor en México.

Un sistema embebido es un sistema electrónico diseñado específicamente para realizar una determinada función, habitualmente formando parte de un sistema de mayor entidad. La característica principal es que emplea para ello un procesador digital (CPU) en formato microprocesador, microcontrolador o un procesador digital de señales (DSP del inglés "*Digital Signal Processor*"), lo que le permite aportar «inteligencia» al sistema anfitrión al que ayuda a gobernar y del que forma parte [2].

La importancia que tiene la domótica para la optimización de los hogares abarca todas las fases de la tecnología del hogar inteligente marcado una gran diferencia implicando un cambio o adaptación de casas utilizando sistemas de seguridad, sistemas ahorradores de energía, alarmas o cámaras de circuito cerrado, tomando en cuenta que es un conjunto de tecnologías que se encuentra actualmente en pleno desarrollo, teniendo orígenes en la automatización industrial, por lo que hoy en día se puede encontrar casas totalmente equipadas con este tipo de tecnología que además de aumentar el confort, proporciona una mayor seguridad en la vivienda y permite crear un uso más eficiente de la energía [3].

2. Metodología

El proceso *Design Thinking* (DT) fue planteado a finales de la década de los 1980 por David Kelley, y posteriormente fue conceptualizado y masificado por Tim Brown, cofundador y presidente de la empresa Ideo. Esta metodología permite desarrollar productos basados en las necesidades del usuario final, con el fin de obtener productos funcionales para solucionar necesidades reales del mercado, el proceso general consta de cinco fases: empatizar, definir, idear, prototipar y probar, como se muestra en la Figura 2 [4].

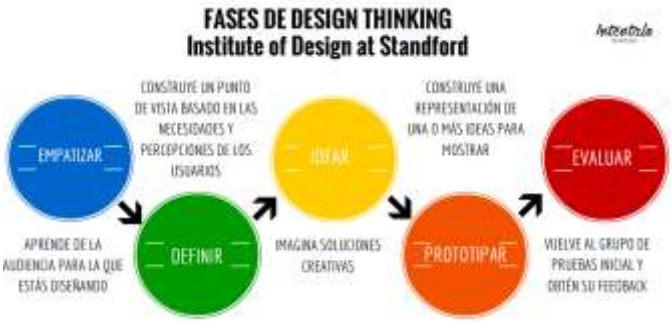


Fig. 2. Fases de desarrollo de metodología *Design Thinking*.

El objetivo general de este proyecto es implementar un sistema de seguridad en puertas y ventanas en casas hogar véase Figura 3, controlado por contraseña a través de un teclado matricial 4x4, en la Figura 4 se puede apreciar los puertos asignados a filas y columnas para su funcionamiento, delimitando a que el usuario tendrá la información correcta para poder controlarlo, dentro de los objetivos específicos es dar solución a la

problemática de inseguridad de robos a casas hogar, teniendo como muestra los resultados de los porcentajes de carpetas de investigación en el año 2020.

Fig. 3. Prototipo de sistema de seguridad visto en 3D.

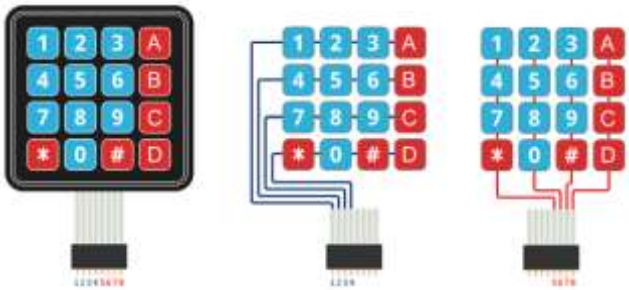


Fig. 4. Esquema del teclado matricial 4x4 y señalización de conexión por filas (líneas color azul) y columnas (líneas color rojo) [5].

Para el desarrollo del proyecto se iniciará por cumplir los puntos a continuación:

- **Necesidad**: desarrollar una herramienta que aumente la seguridad en accesos en casas hogar, alertas al momento de ingresar incorrectamente la contraseña y mostrar en pantalla el estado en que se encuentra abierto, cerrado o error (contraseña incorrecta) véase Figura 5.
- **Objetivo**: diseñar e implementar un sistema que permita el cierre y si es alterado notificar por sonido, indicando apertura, cierre o error, indicando al usuario final el estado en que se encuentra, aumentando la seguridad.
- **Definición del problema**: el problema de inseguridad en México va en aumento aunado a asaltos o robos de automóviles actualmente se tiene robos a hogares.
- **Justificación**: el valor agregado que brinda este sistema a los usuarios finales es incrementar la seguridad a hogares tanto en zonas con índices altos de delincuencia y reincidencia como en las zonas con porcentajes bajos.

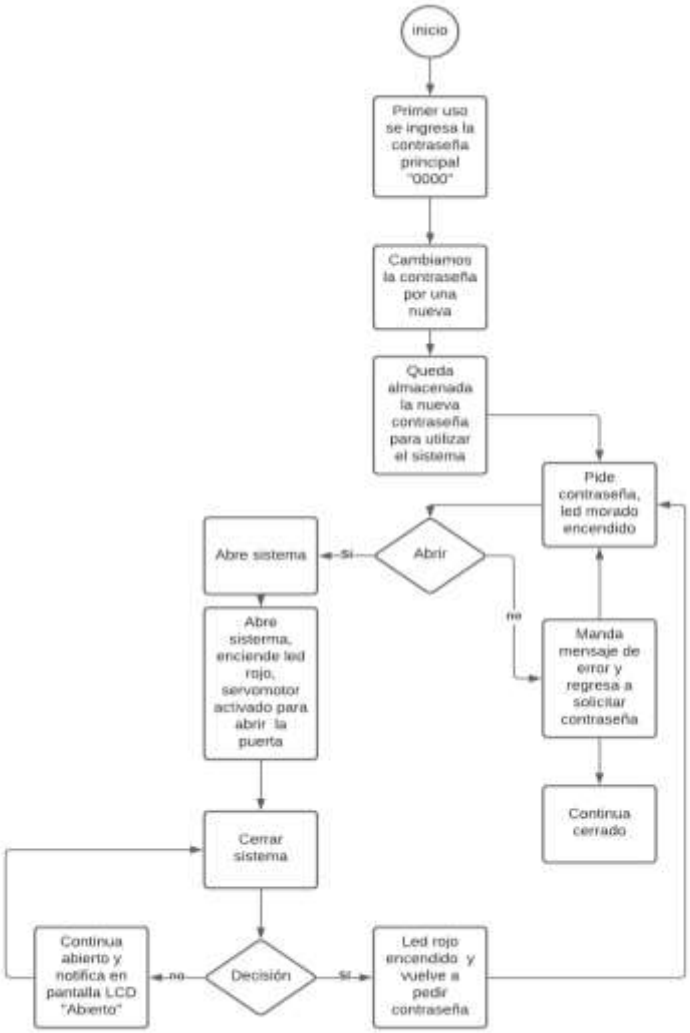


Fig. 5. Diagrama de flujo del funcionamiento del Sistema de seguridad.

3. Implementación y desarrollo

Para el desarrollo de este proyecto se realizó un prototipo que controla la apertura y cierre de accesos en hogares, mediante el ingreso de una contraseña de 4 dígitos utilizando un teclado matricial 4x4 (ver Figura 4), será una serie de números por default de cuatro ceros “0000” pero esta se podrá modificar por el usuario, solamente se puede cambiar cuando se abra la cerradura, que mediante validaciones permite abrir o cerrar utilizando un servomotor [6], el funcionamiento se deriva del movimiento de 30°

grados o 90° grados, ese moviendo permitirá el control de abierto o cerrado como se muestra en la Figura 5, conectado al puerto 11 del Arduino.

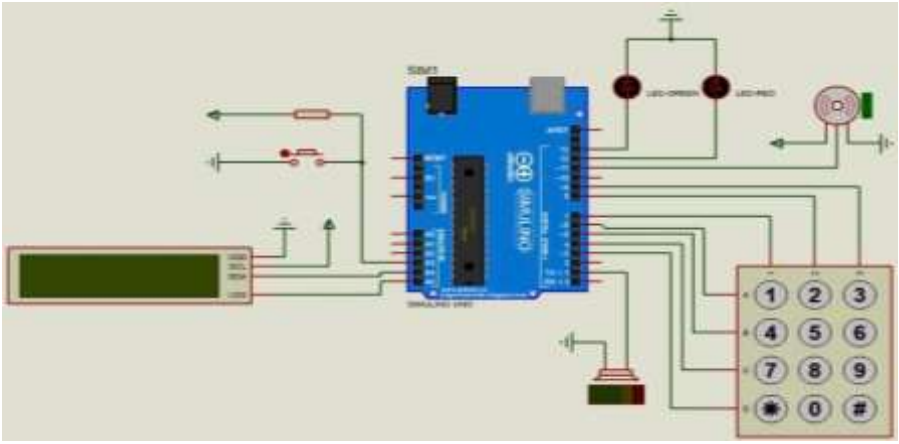


Fig. 6. Diagrama del prototipo Sistema de seguridad para puertas y ventanas, conexión de LCD, Led, servomotor, speaker y teclado matricial.

La apertura y cierre se indican por medio de un Led conectados en los puestos 13 y 12 como se observa en el diagrama de la Figura 6, el led morado indica que se encuentra cerrado ver Figura 7, mientras que el rojo indica que está abierto como se muestra en la Figura 8, se utilizó un *push button* para poder añadir una nueva contraseña que de igual forma será de cuatro dígitos, si un usuario ajeno o al mismo usuario se le olvida o no sabe la contraseña activará la alarma dando aviso que alguien ajeno quiere ingresar, indicado que la contraseña es incorrecta además que encenderá un led rojo indicando la contraseña errónea [7].

Fig. 7. Prototipo en estado cerrado indicado mediante la LCD y el led encendido de color morado encendido.

Para la visualización de los estados en los cuales se encuentra el sistema (cerrado o abierto), se hará uso de una LCD de 16 filas por 2 columnas, definiendo a una LCD como: *Liquid Crystal Dysplay* o pantalla de cristal líquido es un dispositivo empleado para la visualización de contenidos o información de una forma gráfica, mediante caracteres, símbolos o pequeños dibujos dependiendo del modelo.

Al ser de 16x02, se refiere al número de caracteres de largo (16) y por ancho (2), un módulo I^2C [8], véase Figura 10, para la optimización de los pines de la pantalla LCD conectándolo según corresponde (SDA a A4 y VSS a A5) como se muestra en la Figura 4.

De igual manera tendrá uso de un *buzzer* [9], el cual cumplirá con la función de alertar cuando se tenga un acceso incorrecto comenzará a sonar, conectado al puerto 1 y a negativo.

Fig. 8. Prototipo en estado abierto indicado mediante la LCD y el LED encendido de color rojo.

El esquemático del diseño gráfico, muestra las vistas por dentro donde se marca el sistema de seguridad montado junto con el servomotor para hacer el control de apertura u cierre, de igual manera la vista de lado frontal, mostrando el espacio que estaría ocupando, y una tercer vista de la parte exterior para visualizar como sería el esquemático por dentro y fuera, dando mayor factibilidad de uso, instalado a una altura de 1.80 cm del piso, con la finalidad de mantener el control y manejo por adultos y no esté al alcance de los niños evitando un mal uso como se presenta en la Figura 9, el prototipo se desarrolló en *3D Builder* para realizar el maquetado.

Fig. 9. Maquetado del sistema ya instalado en realizado en *3D Builder*.

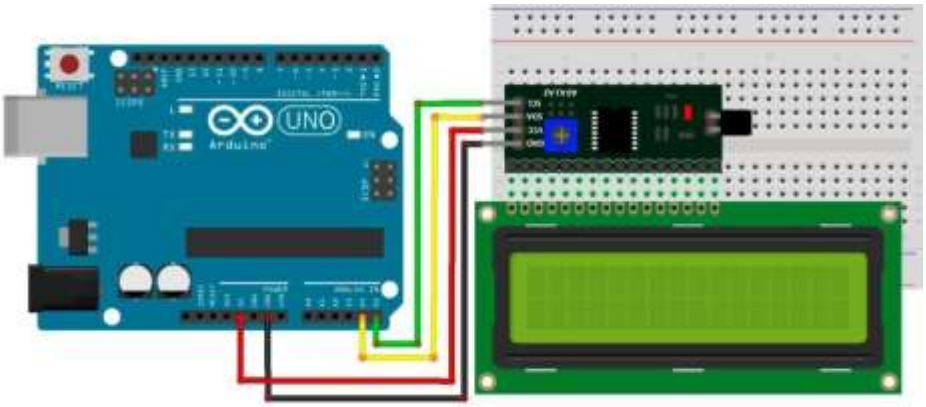


Fig. 10. Simulación de conexión de Arduino a I^2C para reducir el uso de puertos de conexión de una LCD**.**

4. Conclusiones

Se diseñó un sistema de seguridad para puertas y ventanas permitiendo tener un mayor control en los accesos principales notificando al usuario en el estado en que se encuentra mediante la LCD e indicadores led, el sistema permite almacenar una contraseña de 4 caracteres para poder abrir o cerrar, se realizó la programación en Arduino IDE para el funcionamiento del sistema y se realizó un prototipo en físico basado en el diagrama estructurado en Proteus, observando una funcionalidad correcta al ingresar la contraseña permitiendo la apertura y posteriormente el cierre y la activación del *buzzer* en caso de ser incorrecta la serie de números ingresados, permitiendo realizar cambios de clave mediante un *push button* solamente cuando se encuentra en estado de apertura.

Agradecimientos. Los autores agradecen al Tecnológico de Estudios Superiores de Ecatepec, en específico a la división de informática por el apoyo brindado.

Referencias

1. Instituto de Información Estadística y Geográfica de Jalisco (IIEG): Reporte mensual robo casa marzo 2020. (26 mayo 2020) [En línea]
2. Espinosa Meléndez, O. J., Sánchez Díaz, J. J., Hernández Lara, D., Juárez Velázquez, E. T., Téllez Torres, D. J., Pérez Sánchez, M. M.: Diseño conceptual de un sistema embebido para la toma de asistencia en aulas. *Congreso Argentino de Sistemas Embebidos*, vol. 1, pp. 171–173 (2020)
3. Calvo Torres, F. J.: Análisis y diseño de un red domótica para viviendas sociales. Valdivia, Chile: Universidad Austral de Chile (2014)
4. De La Vega Gaytan, M. F., Robles Niño, R. D., García Quijano, B. A., Flores Palmeros, P., Hernández Lara, D., García Hernández, A.: Diseño conceptual de un sistema biométrico para casilleros. Congreso Argentino de Sistemas Embebidos, vol. 1, p. 168 (2020)
5. Llamas, L.: Usar un teclado matricial con Arduino. (2016) [En línea]
6. García González, A.: ¿Qué es y cómo funciona un servomotor? (2016) [En línea].
7. Lima Ortega, E. J., Espillico Condori, J. LL: Diseño e implementación de un sistema integral de seguridad, controlado y monitoreado en forma local y remota mediante las redes de comunicación para las agencias de caja rural. Puno, Perú: Universidad Nacional del Altiplano, pp. 15–16 (2015)
8. Geekk: LCD 16×2 por I2C con Arduino usando solo dos pines. (2017) [En línea]
9. Llamas, L.: Reproducir sonidos arduino buzzer pasivo altavoz. (2016) [En línea].

ISSN 1870-4069

Supervised Machine Learning Application for Developing a Predictive Model of the Monthly Phase of the Pacific Decadal Oscillation

Indalecio Mendoza Uribe

Instituto Mexicano de Tecnología del Agua,
Mexico

indalecio_mendoza@tlaloc.imta.mx

Abstract. In this work, supervised machine learning was applied, using regression trees, to develop a predictive model of the monthly phase of the Pacific Decadal Oscillation. This oscillation is associated with the alteration of weather patterns, mainly in the North Pacific and southwestern North America. As characteristics, the records of the PDO phase of the 24 months prior to the forecast target month were used. The predictive model developed presented an acceptable capacity to estimate the monthly phase of the PDO. This according to the performance evaluation statistics corresponding to the Mean Absolute Error, Maximum Error, Mean Quadratic Error and Pearson's Correlation, which obtained ranges of [0.55,1.07], [1.58,3.29], [0.55,1.82] and [0.30,0.74] respectively for 20% of test data for the period 1854-2020.

Keywords: Artificial intelligence, climate, regression trees.

1 Introduction

In climatology, machine learning has great potential, especially in phenomena of long temporal development, as is the case of the Pacific Decadal Oscillation (PDO). PDO is mainly characterized by changes in sea surface temperature (SST) in the Pacific Ocean over 20° north latitude, as well as variation in sea level pressure and wind patterns. The study of the PDO has gained relevance in recent years due to its association with the alteration of weather patterns, mainly in the North Pacific and southwestern North America [1, 2, 3, 4]. Alterations in the climate have significant socioeconomic impacts, especially in countries that base their development on the management of their natural resources [5].

In the area of artificial intelligence, various machine learning techniques have been applied to understand, describe and predict the behavior of natural phenomena. Ovando et al. [6] developed a model based on neural networks to predict the occurrence of frost in Argentina, based on meteorological data of temperature, relative humidity, cloud cover, wind direction and speed. On the other hand, Téllez-Valero et al. [7] developed a system based on machine learning methods that improves the acquisition of data from

natural disasters, the system automatically populates a database of natural disasters with information extracted from online newspaper news. In addition, Haro-Rivera [8] applied a decision tree to identify predominant meteorological variables in the province of Chimborazo, Ecuador. Finally, in this list of examples, Suárez et al. [9] analyzed the meteorological phenomenon called DANA, which caused serious floods, human losses, economic and infrastructure damage in the southeast of Spain during the month of September 2019, studying the phenomenon from the perspective of data analysis.

Machine learning is a data analysis technique that gives computers the ability to learn from experience without relying on a given equation as a model. These algorithms look for natural patterns in the data that generate knowledge. Algorithms adaptively improve their performance as the number of samples available for learning increases. In a general way, we can classify machine learning techniques as supervised and unsupervised.

A supervised learning algorithm takes a set of known data (inputs) and known responses for this data (outputs) to train a model that can generate reasonable predictions in response to new data. Supervised learning uses classification and regression techniques to develop predictive models. In comparison, unsupervised learning looks for hidden patterns or intrinsic structures in the data. Used to infer information from data sets consisting of input data with no labeled responses. Among the most common unsupervised learning techniques are neural networks [10], k-means [11], among other.

The objective of this work was to apply supervised machine learning through regression trees to develop a predictive model of the monthly phase of the Pacific Decadal Oscillation. As characteristics, the records of the PDO phase of the 24 months prior to the target month of prognosis were used.

2 Method

The development of the predictive model was carried out by applying three procedures. First, the historical data set of the monthly value of the PDO was obtained for the period 1854-2020. These data were organized by month and grouped into training and test data. Second, for each month of the year the regression tree corresponding to the predictive model was generated with the training data. Third, the monthly predictive models were applied on the test data sets. The results were evaluated using three continuous error measurement metrics and one of correlation.

2.1 Dataset

The PDO is a pattern of anomalies of the SST, this fluctuation oscillates between -4 and 4 degrees centigrade, corresponding to the cold and warm phase respectively. The PDO values indicate the variation of the SST with respect to the historical average. The data was obtained from National Oceanic and Atmospheric Administration through the URL https://www.ncdc.noaa.gov/teleconnections/pdo/data.csv. The data set corresponds to the monthly deviation of the SST for the period 1854-2020 (see Fig. 1). For

each forecast month (label) the values of the previous 24 months were assigned as characteristics. The characteristics and labels for each month of the year were grouped in separate files to facilitate their processing.

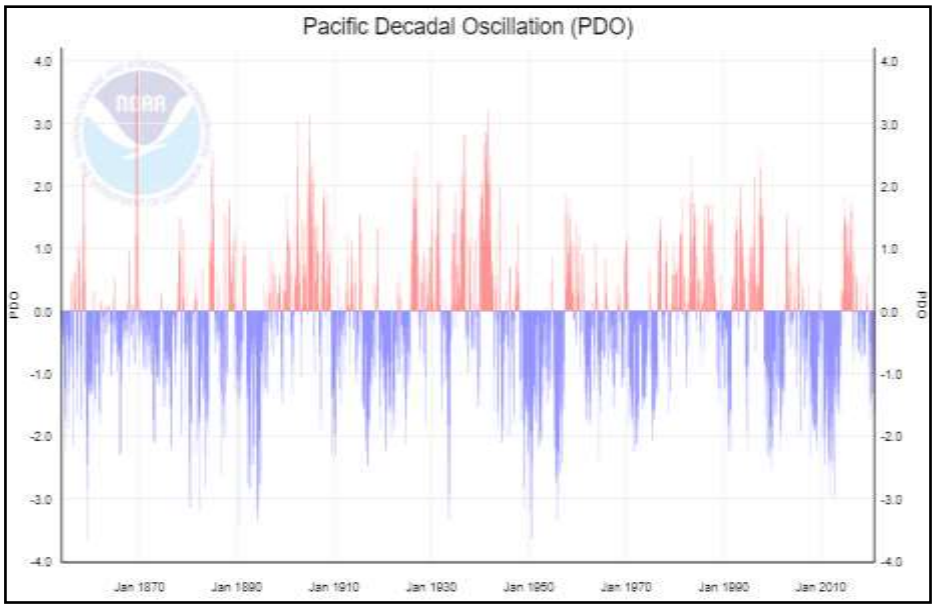


Fig. 1. Monthly anomaly of the Pacific Decadal Oscillation for the period 1854-2020 [12].

Machine learning consists of learning some properties of a data set and then verifying those properties with another data set. A common practice in machine learning is to evaluate an algorithm by dividing the data into two subsets. The majority set is dominated by training data, from which the algorithm learns some properties. While the second set of data is called test data, with which the ability of the model to predict through the learned properties is verified. For this study, the training and test data set were divided into a proportion of 80 and 20% respectively.

2.2 Generation of the Predictive Model

For each month of the year, the regression tree corresponding to the predictive model was generated. Each predictive model was trained with 80% corresponding training data.

Classification and regression trees (CART) were developed by Breiman et al. [13]. Tree models where the target variable can take a finite set of values are called classification trees. On the other hand, trees where the target variable can take continuous values are called regression trees.

Let Y be the response variable and x be the vector with the set of predictor variables, the problem corresponds to establishing a relationship between Y and x in such a way

that it is possible to predict *Y* based on the values of *x*. Mathematically looking for probability $P(Y \mid x_1, x_2, \ldots, x_k)$.

The construction of the tree is done following a recursive binary division approach, let *N* be the number of data and N_j the number of cases in class *j*.

The probability that a case is in class *j* given that it was located in the terminal node *t*, is given by the Eq. 1.

$$P(j \mid t) = \frac{P(j,t)}{P(t)} = \frac{N_j(t)}{N} \tag{1}$$

and comply with:

$$\sum P(j \mid t) = 1. \tag{2}$$

Thus, the set of $P(j|t)$ are the relative proportions of the cases in class *j* at node *t* [8].

To obtain the optimal tree, evaluate each subdivision among all possible trees, get the root node and the subsequent ones, the algorithm must measure the predictions achieved and evaluate them to select the best one. Fig. 2 shows a simplified form of a regression tree.

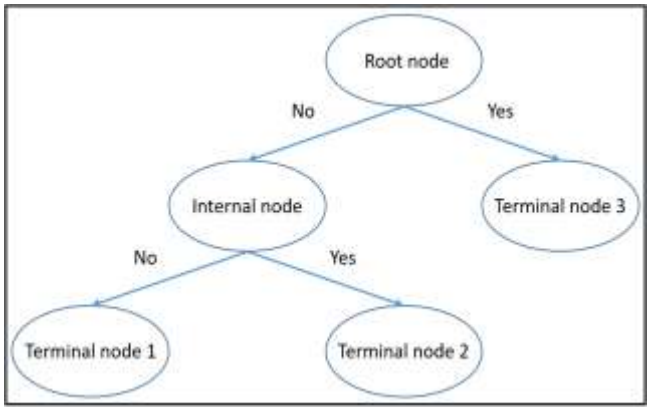


Fig. 2. Simplified form of a regression tree.

In this study, machine learning was applied through the *Scikit-Learn* library of the Python programming language, which integrates a wide range of machine learning algorithms for supervised and unsupervised problems [14].

Specifically, the *tree.DecisionTreeRegressor* method was used to create the instance of the predictive model; *train_test_split* to divide the training/test data set; *mean_absolute_error, mean_squared_error y max_error* to measure mean absolute error, mean square error and maximum error respectively; finally, the function *plot_tree* was used to graph the regression trees.

2.3 Statistical Validation of the Predictive Model

Monthly predictive models were applied on the corresponding test data sets. For the evaluation of the monthly predictive model of the PDO phase, three continuous error measurement metrics and Pearson's correlation were used. These metrics are recommended for evaluating forecasts of a deterministic nature. These metrics are described below.

The Mean Absolute Error (MAE) measures the magnitude of the errors in a set of predictions, regardless of their direction [15, 16]. It corresponds to the average of the absolute differences between the prediction and the observation where all the individual differences have the same weight (Eq. 3):

$$\mathrm{MAE} = \sum_{i=1}^{n} \frac{|P_i - O_i|}{n}, \tag{3}$$

where P_i is the prediction value at position i, O_i is the value observed at position i and n is the sample size.

The Maximum Error (ME) allows to identify the largest absolute value of the observed error between the prediction and the observation (Eq. 4). It belongs to the set of objective functions used for the calibration of models [17]:

$$\mathrm{ME} = \sum_{i=1}^{n} \max\{|P_i - O_i|\}. \tag{4}$$

The Root Mean Square Root (RMSE) measures the mean magnitude of the error. Corresponds to the square root of the average of the squared differences between the prediction and the observation, therefore this measure has been used in the evaluation of forecasting models [18, 19]. Amplifies and penalizes with greater force those errors of greater magnitude (Eq. 5):

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(P_i - O_i)^2}. \tag{5}$$

Pearson's Correlation, denoted as r (Eq. 6), is a normalized measure widely used to establish relationships between two continuous quantitative variables [20, 21]. It allows to show the joint variability and therefore to typify what happens with the data. The coefficient can score values ranging from -1.0 to 1.0 and is interpreted as follows: values close to 1.0 indicate that there is a strong association between the variables, that is, they increase or decrease in the same direction.

On the other hand, values close to -1.0 indicate that there is a strong negative association between the variables, that is, as one variable increases, the other decreases. A value of 0.0 indicates that there is no correlation or it is a null correlation [22].

$$r = \frac{\sum_{i=n}^{n}(P_i - \bar{P})(O_i - \bar{O})}{\sqrt{\sum_{i=n}^{n}(P_i - \bar{P})^2}\sqrt{\sum_{i=n}^{n}(O_i - \bar{O})^2}}, \quad (6)$$

where $\bar{P}$ is the mean value of the predictions and $\bar{O}$ is the mean value of the observations.

3 Results

For the creation of the monthly predictive models based on regressive trees, the constructor of the *DecisionTreeRegressor* class was used. Table 1 lists the parameters used during the creation of the predictive model with which the best results were obtained.

Table 1. Predictive model creation parameters.

Parameter	Value	Description
criterion	mse	Function to measure the quality of the division.
splitter	best	Strategy used to choose the division at each node.
max_depth	None	Maximum depth of the tree. None indicates that nodes are expanded until all sheets are pure or until all sheets contain less than *min_samples_split* samples.
min_samples_split	2	The minimum number of samples required to divide an internal node.
min_samples_leaf	1	The minimum number of samples required to be in a leaf node.
max_features	12	The number of features to consider when looking for the best division.
random_state	5	Controls the randomness of the estimator. To obtain a deterministic behavior during the setting *random_state* must be set to an integer.

As part of the training, the algorithm identifies the impact on the prognosis of each of the characteristics. As can be seen in Table 2, in general, with 12 characteristics, more than 90% importance is obtained in the forecast.

These 12 characteristics are not the same for all months of the year, therefore, in the training stage, the 24 characteristics are initially considered, but the algorithm is instructed to only select the 12 most relevant characteristics. This reduction in dimensions

allows the algorithm to be optimized by eliminating characteristics that do not contribute to the forecast.

Table 2. Percentage of importance by characteristics for monthly predictive models.

		Month											
		Jan	**Feb**	**Mar**	**Apr**	**May**	**Jun**	**Jul**	**Aug**	**Sep**	**Oct**	**Nov**	**Dec**
Characteristics	**1**	1	0	1	0	3	8	4	2	0	1	3	1
	2	0	1	1	0	0	3	1	0	0	2	4	7
	3	1	0	3	1	1	2	1	1	0	3	3	1
	4	0	1	2	0	2	0	0	3	5	4	0	0
	5	5	2	1	3	2	0	2	5	1	2	0	0
	6	0	1	1	0	1	3	0	2	0	2	1	0
	7	1	0	0	4	3	2	1	2	0	0	1	0
	8	0	2	0	0	2	0	1	1	3	0	1	0
	9	2	1	1	1	0	2	2	1	5	0	1	6
	10	4	3	2	0	3	1	0	3	2	2	1	3
	11	3	0	2	0	0	0	1	2	0	0	2	4
	12	0	1	4	0	0	0	1	0	3	2	4	2
	13	8	6	0	0	0	0	1	4	0	2	0	1
	14	0	1	1	2	1	0	2	0	1	0	1	0
	15	6	0	3	1	5	0	1	1	1	2	1	0
	16	3	1	2	1	1	0	0	2	3	1	0	0
	17	0	3	1	0	1	2	0	0	1	7	5	7
	18	1	2	1	0	0	3	0	5	0	1	1	11
	19	2	1	1	5	1	0	2	2	3	4	4	2
	20	0	3	1	1	2	6	3	3	0	2	1	2
	21	5	1	0	5	1	0	3	2	2	6	0	2
	22	0	1	1	3	2	2	1	37	7	1	1	0
	23	39	49	56	50	51	35	58	14	36	38	41	40
	24	19	20	15	23	18	31	15	8	27	18	24	11

Algorithm 1 presents in a simplified way the sequence of steps to divide the data into the training/test subsets, feed the classifier (predictive model) with the training data, apply the classifier on the test data, calculate model performance evaluation metrics, graphing and data storage. Clarification is made that the algorithm does not detail the modules of *dataReadingMonth()* and *graphingStorage()*.

Algorithm 1: Simplified sequence to generate, train, apply and validate the monthly predictive model

```
for month in range(0,12):
    totalCharacteristics, totalLabels = dataReadingMonth(month)
    trainingCharacteristics, testCharacteristics, trainingLabels, testLabels =                \
            train_test_split(totalCharacteristics, totalLabels,train_size=0.80,               \
            test_size=0.20, random_state= 5)
    # Creation of the instance (object) of type DecisionTreeRegressor (predictive model)
    predictiveModel = tree.DecisionTreeRegressor(criterion = 'mse', splitter = 'best',        \
             max_depth = None, min_samples_split = 2, min_samples_leaf = 1,                   \
             max_features = 12, random_state=5)
    # Feed the classifier with the training data (train the predictive model)
    predictiveModel.fit(trainingCharacteristics,trainingLabels)
    # Apply the predictive model to the test data set
    predictions = predictiveModel.predict([testCharacteristics])
    predictedLabels = predictions[0]
    # Calculate MAE, ME, RMSE and r performance metrics
    mae     = round(mean_absolute_error(testLabels,predictedLabels),2)
    me      = round(max_error(testLabels,predictedLabels),2)
    rmse    = round(mean_squared_error(testLabels,predictedLabels),2)
    pearson = sc.pearsonr(testLabels,predictedLabels)
    r = round(pearson[0],2)
    storageGraph(month,predictiveModel,mae,me,rmse,r)
```

Figures 3 and 4 show arbitrarily the trees corresponding to the predictive models for the months of June and December, respectively.

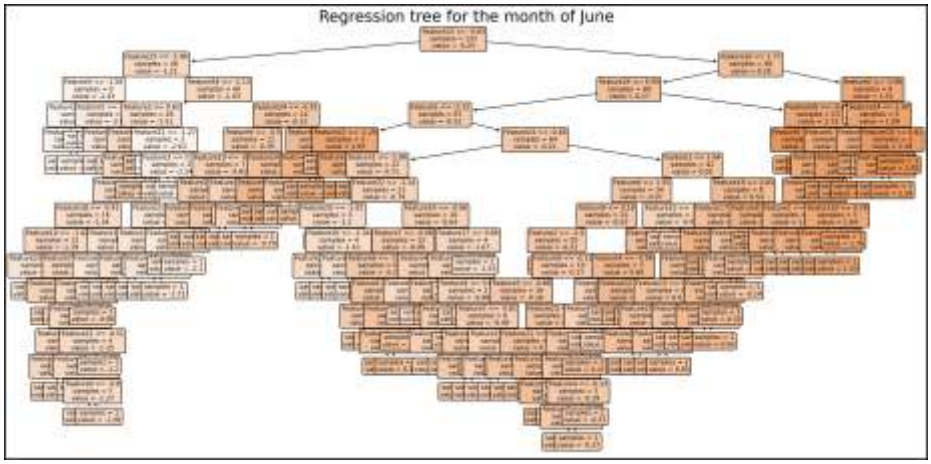


Fig. 3. Regression tree for the month of June. The predictive model was trained with 80% of data from the period 1854-2020. The strongest fill color indicates the majority class for classification.

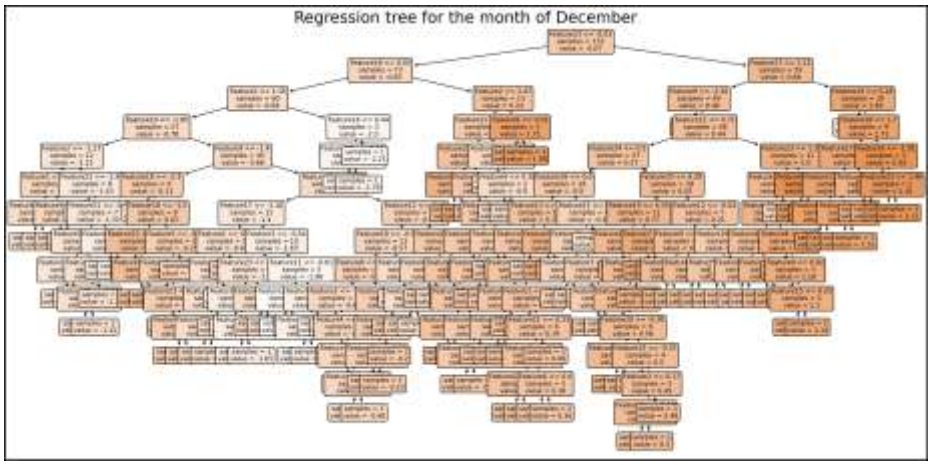


Fig. 4. Regression tree for the month of December. The predictive model was trained with 80% of data from the period 1854-2020. The strongest fill color indicates the majority class for classification.

Monthly predictive models were applied for 20% of test data. Table 3 shows the results of the four statistical metrics applied by the monthly predictive model. Besides that, Fig. 5 shows the dispersion diagrams with the comparison between the observed and predicted data.

Table 3. Result of the statistical metrics of the monthly predictive models.

Target Month	MAE	ME	RMSE	r
January	0.79	2.49	1.03	0.59
February	0.66	2.09	0.70	0.64
March	0.61	2.44	0.66	0.72
April	0.74	1.58	0.74	0.74
May	0.86	2.52	1.13	0.62
June	0.64	1.79	0.62	0.77
July	1.01	2.47	1.44	0.55
August	1.07	3.29	1.82	0.30
September	1.01	2.83	1.58	0.38
October	0.69	1.60	0.68	0.74
November	0.55	1.75	0.55	0.77
December	0.89	3.53	1.31	0.47

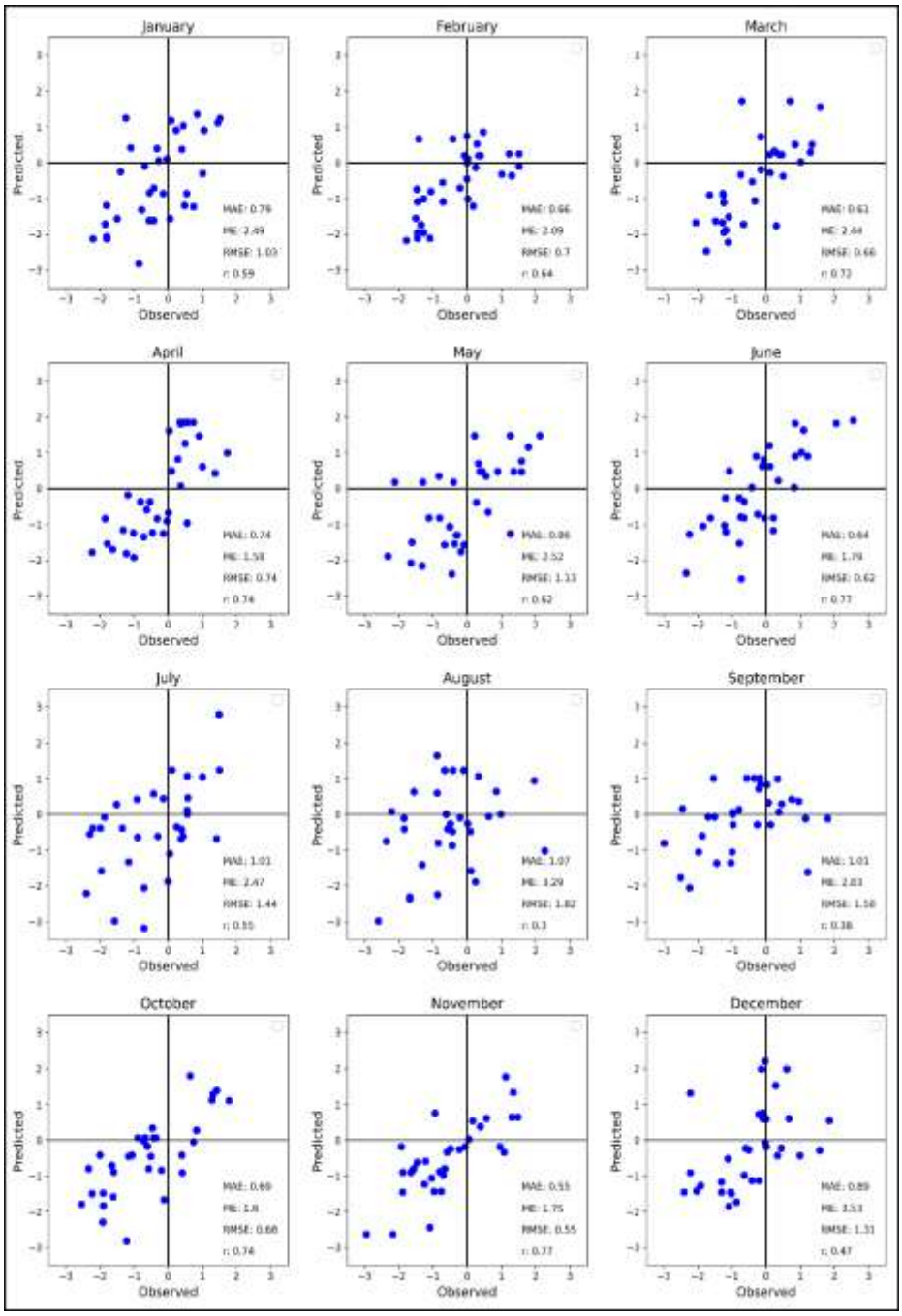


Fig. 5. Monthly dispersion diagrams between observed and predicted values for 20% of test data for the 1854-2020 period.

4 Conclusions

Of the 24 characteristics considered, it was identified that characteristic 23 in eleven months and characteristic 22 in the month of July, predominated as root node in the trees of the predictive models, that is, these characteristics have a greater impact on forecasts. In addition, it was distinguished that in 12 characteristics more than 90% of importance is obtained in the prognosis.

The predictive model developed using machine learning presented an acceptable capacity to estimate the monthly phase of the PDO. This according to the results of the performance evaluation statistics MAE, ME, RMSE and r obtained for 20% of test data, with ranges of [0.55, 1.07], [1.58, 3.29], [0.55, 1.82] y [0.30, 0.74] respectively. Therefore, it is considered that the predictive model developed can constitute a reference forecasting tool, but not an exact one.

As future work, it is proposed to continue with the validation and adjustment of the predictive model for its application in larger time windows, such as for seasonal forecast (3 months), or even annual forecast.

Regarding the functionality of the Scikit-Learn library, this turned out to be docile to implement and very efficient in its performance. The computational cost required for the training and testing of the predictive model was of the order of seconds on a personal computer.

References

1. Mantua, N.J., Hare, S.R.: The Pacific Decadal Oscillation. Journal of Oceanography, 58, 35–44 (2002). Doi: https://doi.org/10.1023/A:1015820616384
2. Cayan, D.R., Dettinger, M.D., Diaz, H.F., Graham, N.E.: Decadal variability of precipitation over western North America. Journal of Climate, 11, 3148-3166 (1998). Doi: https://doi.org/10.1175/1520-0442(1998)011<3148:DVOPOW>2.0.CO;2
3. Higgins, R.W., Leetmaa, A., Xue, Y., Barnston, A.: Dominant factors influencing the seasonal predictability of U.S. precipitation and surface air temperature. Journal of Climate, 13(22), 3994–4017 (2000). Doi: https://doi.org/10.1175/1520-0442(2000)013<3994:DFITSP>2.0.CO;2
4. Gutzler, D.S., Kann, D.M., Thornbrugh, C.: Modulation of ENSO-based long-lead outlooks of Southwest U.S. Winter precipitation by the Pacific Decadal Oscillation. Weather and Forecasting, 17, 1163–1172 (2002).
5. Méndez-González, J., Ramírez-Leyva, A., Zárate-Lupercio, A., Cavazos-Pérez, T.: Teleconexiones de la Oscilación Decadal del Pacífico (PDO) a la precipitación y temperatura en México. Investigaciones Geográficas, 73, 57–70 (2010).
6. Ovando, G., Bocco, M., Sayago, S.: Redes neuronales para modelar predicción de heladas. Agricultura Técnica, 65(1), 65–73 (2005). Doi: http://dx.doi.org/10.4067/S0365-28072005000100007
7. Téllez-Valero, A., Montes, M., Villaseñor-Pineda, L.: Using Machine Learning for Extracting Information from Natural Disasters News Reports. Computación y Sistemas, 13(1), 33–44 (2009).
8. Haro-Rivera, S.M.: Árbol de decisión, aplicación con datos meteorológicos. KnE Engineering, 5(2), 37–46 (2020). Doi: https://doi.org/10.18502/keg.v5i2.6217

9. Suárez, L., Alarcon, P.A.: Inteligencia artificial para la comprensión de desastres naturales. Telefónica Digital, España (2020).
10. Mercado-Polo, D., Pedraza-Caballero, L., Martínez-Gómez, E.: Comparación de Redes Neuronales aplicadas a la predicción de Series de Tiempo. Prospectiva, 13(2), 88–95 (2015). Doi: http://dx.doi.org/10.15665/rp.v13i2.491
11. Kanungo, T., Mount, D.M., Netanyahu, N.S., Piatko, C.D., Silverman, R., Wu, A.Y.: An efficient k-means clustering algorithm: analysis and implementation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 24(7), 881–892 (2002). Doi: https://doi.org/10.1109/TPAMI.2002.1017616
12. NOAA (National Oceanic and Atmospheric Administration) Pacific Decadal Oscillation (PDO), https://www.ncdc.noaa.gov/teleconnections/pdo, last accessed 2021/02/22.
13. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: Classification and Regression Trees. Chapman & Hall/CRC, New York (1984). Doi: https://doi.org/10.1201/9781315139470
14. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D.: Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12(85), 2825–2830 (2011).
15. Willmott, C.J., Matsuura, K.: Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. Climate Research, 30, 79–82 (2005).
16. Karamirad, M., Omid, M., Alimardani, R., Mousazadeh, H., Heidari, S.N.: ANN based simulation and experimental verification of analytical four and five parameters models of PV modules. Simulation Modelling Practice and Theory, 34, 86–98 (2013).
17. Gupta, H.V., Sorooshian, S., Yapo, P.O.: Toward improved calibration of hydrologic models: Multiple and noncommesurable measure of information. Water Resources Research, 34(4), 751–763 (1998).
18. González-Leyva, F., Ibáñez-Castillo, L.A., Valdés, J.B., Vázquez-Peña, M.A., Ruiz-García, A.: Pronóstico de caudales con Filtro de Kalman Discreto en el río Turbio. Tecnología y Ciencias del Agua, 6(4), 5-24 (2015).
19. Vázquez, M.: Predicción de series de tiempo usando un modelo híbrido basado en la descomposición wavelet. Comunicaciones estadísticas, 11(2), 257–283 (2018).
20. Restrepo, L.F., González, J.: De Pearson a Spearman. Revista Colombiana de Ciencias Pecuarias, 20, 183–192 (2007).
21. Martínez-Curbelo, G., Cortés-Cortés, M.E., Pérez-Fernández, A.C.: Metodología para el análisis de correlación y concordancia en equipos de mediciones similares. Universidad y Sociedad, 8(4), 65–70 (2016).
22. Anderson, R.B., Doherty, M.E., Friedrich, J.C.: Sample size and correlational inference. Journal of Experimental Psychology: Learning, Memory and Cognition, 34(4), 929–944 (2008). Doi: https://doi.org/10.1037/0278-7393.34.4.929

ISSN 1870-4069

Procesamiento de imágenes infrarrojas para detectar deficiencias de nitrógeno de cultivos en ambientes controlados usando dispositivos móviles

Luis A. Gama-Moreno, Carlos Martínez Hernández, Abel Ramírez Molina, José A. Torres Rangel, José L. Torres Rodríguez

Tecnológico Nacional De México, Campus Tlajomulco, Tlajomulco de Zúñiga, Jal., México

{luis.gm, carlos.mh, abel.rm, jose.tr, jose1.tr}@tlajomulco.tecnm.mx

Resumen. En este artículo se presenta un sistema para la detección de deficiencia de nutrientes en plantas dentro de cultivos controlados (invernaderos) denominado NDVICam. NDVICam está basado en el Índice Diferencial de Vegetación Normalizado (NDVI: Normalized Difference Vegetation Index) el cual es usado para estimar la cantidad, calidad y desarrollo de la vegetación con base a la medición (por medio de sensores remotos) de la intensidad de la radiación de ciertas bandas del espectro electromagnético que la vegetación refleja. Este sistema se basa en imágenes tomadas a cultivos en ambientes controlados, realizando dos tomas del mismo objetivo 1) una imagen en el espectro de colores normal y 2) una segunda foto en el espectro infrarrojo, procesando ambas imágenes es como se obtiene el índice NDVI. Asimismo, se presenta un prototipo de cámara dual para capturar las dos imágenes, tanto en el espectro RGB como en el infrarrojo. Con el uso de NDVICam, los usuarios podrán obtener datos sobre nutrientes (en esta etapa solo Nitrógeno) de sus plantas en cualquier momento y en cualquier lugar sin tener que esperar para su procesamiento o tener que llevar las imágenes a un centro especializado para su procesamiento.

Palabras clave: Procesamiento de imágenes, NDVI, agricultura de precisión, cómputo móvil, espectro infrarrojo.

Infrared Image Processing for Detection of Nitrogen Deficiencies of Crops in Controlled Environments Using Mobile Devices

Abstract. In this paper, a system for the detection of nutrient deficiency in plants within controlled crops (greenhouses) called NDVICam is presented. NDVICam is based on the Normalized Difference Vegetation Index (NDVI) which is used to estimate the quantity, quality, and development of vegetation based on the

measurement (through remote sensing) of the radiation's intensity that vegetation reflects. This system is based on images taken to crops in controlled environments, taking two shots to the same target: 1) an image in the normal color spectrum and 2) a second photo in the infrared spectrum, processing both images is how the NDVI index is obtained. Furthermore, a dual-camera prototype is presented to capture the two images, both in the RGB and infrared spectrum. With NDVICam, users will be able to obtain data (Nitrogen only at this moment) from their plants anytime, anywhere without having to wait for processing or having to take the images to a specialized center for processing.

Keywords: Image processing, NDVI, precision agriculture, mobile computing, infrared spectrum.

1. Introducción

1.1. Procesamiento de imágenes

El procesamiento de imágenes digitales ha crecido vertiginosamente desde el momento en que se crearon dispositivos tecnológicos para captar y manipular grandes cantidades de información espacial en forma de matrices de valores, y de esta manera se han ido perfeccionando técnicas que se aplican a las imágenes digitales con el objetivo de mejorar la calidad o facilitar la búsqueda de información [5]. Entre estas técnicas están el 1) filtrado, que es el conjunto de técnicas englobadas dentro del preprocesamiento de imágenes cuyo objetivo fundamental es obtener, a partir de una imagen origen, otra final cuyo resultado sea más adecuado para una aplicación específica mejorando ciertas características de la misma que posibilite efectuar operaciones del procesado sobre ella; 2) procesamiento de punto, que consiste en la mejora de la imagen considerando los métodos de procesamiento que se basan solo en la intensidad de píxeles individuales; 3) reconocimiento de patrones, para el reconocimiento de caracteres ópticos usando algoritmos de OCR (Optical Character Recognition), entre otros.

Los beneficios del procesamiento de imágenes digitales han permitido grandes avances en diversos campos tales como en la arqueología, donde se procesan las imágenes para una observación remota de la superficie de la tierra. Con la creación de la tecnología LiDAR (por sus siglas en inglés Light Detection and Ranging o Laser Imaging Detection and Ranging) es un dispositivo que permite determinar la distancia desde un emisor láser a un objeto o superficie utilizando un haz láser pulsado [2].

La distancia al objeto se determina midiendo el tiempo de retraso entre la emisión del pulso y su detección a través de la señal reflejada. En general, la tecnología LíDAR tiene aplicaciones en geología, sismología y física de la atmósfera. También se investiga su uso en vehículos, especialmente los autónomos [2]. En la [3] se muestra la estructura maya más grande y más antigua de México recién descubierta gracias al sensor LiDAR, se trata de una enorme plataforma elevada que se estima fue construida entre los años 1000 y 800 A.C., y se ubica en la región de Aguada Fénix cerca de la frontera con Guatemala, dentro del estado de Tabasco en México. En la [4] se muestran las diferentes formas de funcionamiento de la tecnología LíDAR. En la medicina; en

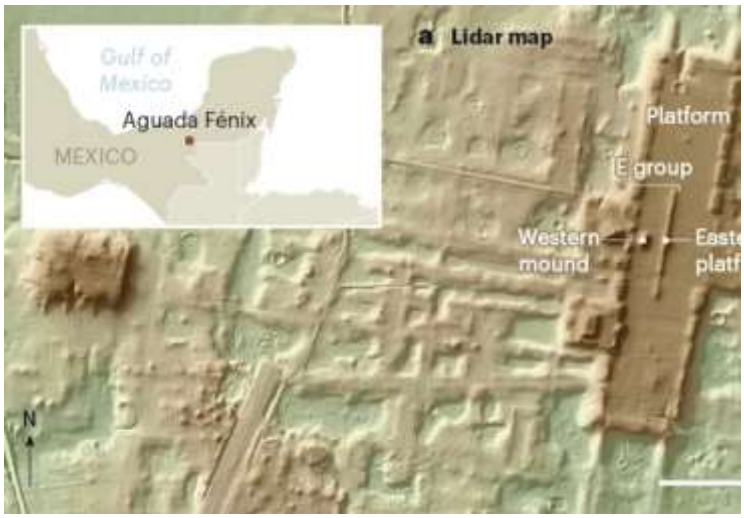


Fig. 1. Zona arqueológica maya descubierta con Lidar.

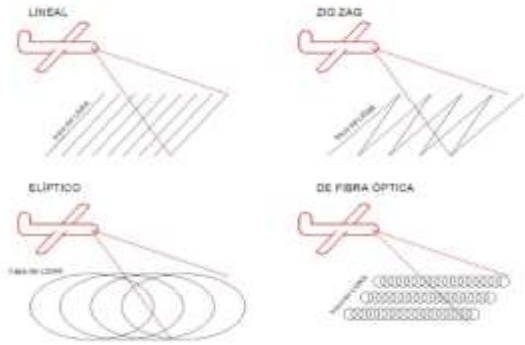


Fig. 2. Tipos de LiDAR.

1971 se produjo la primera imagen del TAC (Tomografía Computarizada); por medio de la cual haciendo uso de los rayos X se obtienen imágenes del interior del organismo y de esta forma es posible detectar desde un tumor hasta daños en la columna vertebral [1].

En el campo climatológico, se han tomado imágenes con satélites y se les ha hecho su respectivo procesamiento, detectando todo tipo de fenómenos naturales, como huracanes, cambios de clima, tornados, entre otros. La base de datos del consumo global de energía o base de datos de "Luces de noche en el mundo" se ha obtenido a través de procesamiento de imágenes de la banda infrarroja; logrando así una visualización leve de las luces encendidas en países, ciudades y pueblos [9]. También se ha utilizado el procesamiento digital de imágenes en el reconocimiento de la huella digital de una persona, en la autenticidad de los billetes, entre otros. Y un sector donde

estas tecnologías han creado un gran impacto es el de la agricultura con la creación del concepto "Agricultura de Precisión" [3].

1.2. Agricultura de precision

La Agricultura de Precisión (AP) consiste en la aplicación de un conjunto de técnicas, apoyadas por equipamiento de alta tecnología dentro de las cuales destacan, los Sistemas de Posicionamiento Global (GPS), sensores remotos, imágenes aéreas y/o satelitales junto con Sistemas de Información Geográfico (GIS), el uso de estas tecnologías contribuye a una adecuada toma de decisiones, desde el punto de vista del manejo técnico-productivo, económico y ambiental; con la finalidad de identificar, analizar y modelar la variabilidad espacial y temporal de los cultivos agrícolas para poder manejarla de acuerdo a los objetivos productivos.

De esta manera los avances en la AP han permitido mejorar considerablemente el nivel predictivo, a través de la investigación y desarrollo de los siguientes componentes: 1) La variabilidad espacial de rendimiento y calidad. 2) El uso de la teledetección en la agricultura. 3) La integración del Índice de Vegetación de la Diferencia Normalizada (NDVI) sobre el estatus hídrico de la planta. 4) La integración del NDVI sobre la calidad y el rendimiento y 5) aplicación específica del volumen foliar como indicador de vigor y expresión vegetativa de un campo agrícola.

El NDVI es un índice usado para estimar la cantidad, calidad y desarrollo de la vegetación con base a la medición (por medio de sensores remotos) de la intensidad de la radiación de ciertas bandas del espectro electromagnético que la vegetación emite o refleja [7]. El NDVI se calcula a partir de la luz visible e infrarroja cercana (NIR: Near InfraRed) reflejada por la vegetación. La vegetación sana (ver Fig. 3, izquierda) absorbe la mayor parte de la luz visible que la golpea, y refleja una gran parte de la luz infrarroja cercana.

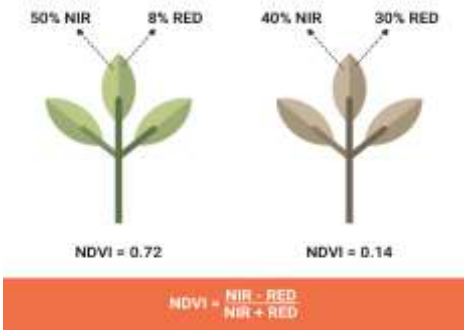


Fig. 3. Niveles de refracción de NIR y Red (color rojo).

La vegetación no saludable o escasa (ver Fig. 3, derecha) refleja más luz visible y menos luz infrarroja cercana. Casi todos los índices satelitales de vegetación emplean esta fórmula de diferencia para cuantificar la densidad del crecimiento de la planta en la Tierra: donde el índice NDVI es igual a radiación infrarroja cercana menos radiación

visible dividida por radiación infrarroja cercana más radiación visible [4]. El resultado de esta fórmula se llama índice de vegetación de diferencia normalizada (NDVI). Escrito matemáticamente, la fórmula es: NDVI = (NIR – RED) (NIR + RED).

Los cálculos de NDVI para un píxel dado siempre dan como resultado un número que va de menos uno (-1) a más uno (+1); sin embargo, ninguna hoja verde da un valor cercano a cero. Un cero significa que no hay vegetación y cerca de +1 (0.8 - 0.9) indica la mayor densidad posible de hojas verdes.

En este campo también se han desarrollado diversos trabajos para ayudar al agricultor. En [8] se construyó un Vehículo Aéreo No-Tripulado (UAV: Unmanned Aerial Vehicle) con un sistema de teledetección ligero, equipado con adquisición y procesamiento de imágenes, lo que resulta en un método simple para obtener información cuantitativa y confiable sobre el crecimiento de los cultivos.

En [10] se presenta una nueva metodología para monitorear la fenología de la vegetación global a través de series de tiempo de datos satelitales.

En [6] se evalúan dos índices de vegetación diferentes de dos nuevos sensores diseñados para el monitoreo de vegetación lanzados en años recientemente: 1) el Espectrómetro de Imágenes de Resolución Moderada (MODIS: Moderate Resolution Imaging Spectrometer) a bordo de los satélites Terra (EOS AM-1) y Aqua (EOS-PM 1), y 2) el Espectrómetro de Imágenes de Resolución Media (MERIS: Medium Resolution Imaging Spectrometer) en el satélite ENVISAT y VEGETATION.

En [7] se describe un sistema para adquisición de fotografías digitales para obtener los canales NIR-green-blue (Near InfraRed) desde vehículos aéreos no-tripulados para el monitoreo de cultivos.

En este artículo se presenta una aplicación diseñada para dispositivos móviles que es capaz de obtener el índice de Vegetación Diferencial Normalizado (NDVI) para conocer el estado de salud de una sección del cultivo, denominada NDVICam.

Esta aplicación es capaz de obtener el índice NDVI al procesar dos imágenes del mismo objetivo (una imagen en formato RGB y otra bajo el espectro infrarrojo), mediante el uso de una cámara modificada adaptada con dos lentes para capturar ambos espectros y así calcular el índice NDVI justo donde se encuentra el objetivo.

NDVICam está desarrollada para las principales plataformas móviles como son Android e iOS así como también para equipos de escritorio como Windows y MacOS. NDVICam les permite a los agricultores, capturar las imágenes de la planta, procesarlas y obtener el índice NDVI, obtiene una interpretación de los índices para determinar el nivel de nitrógeno, además que le permite compartir los resultados con otros agricultores a través de las principales redes sociales.

El resto del artículo está organizado de la siguiente forma. En la sección 2 se describe la metodología para el procesamiento de las imágenes, desde la captura de las imágenes RGB e infrarroja, hasta la generación de los índices NDVI. En la sección 3 se muestra el diseño e implementación de NDVICam así como el resultado e interpretación del proceso de NDVI. Finalmente se presentan las conclusiones.

2. Metodología

2.1. Capturando imágenes

NDVICam se compone de dos elementos principales los cuales son: 1) una cámara dual modificada para capturar imágenes que consta de dos lentes, uno para capturar imágenes del espectro infrarrojo y otro lente normal para capturar los colores RGB, como se ilustra en la Fig. 4(A); y 2) una aplicación para dispositivos móviles desarrollada bajo la plataforma multidispositivo denominada Firemonkey y generada para las principales plataformas móviles tales como Android e iOS, así como para sistemas operativos de escritorio tales como Windows, MacOS y Linux, como se muestra en la Fig. 4(B).

Fig. 4. Elementos que componen la app NDVICam.

2.2. Prototipo de cámara dual

Una cámara térmica o cámara infrarroja es un dispositivo que, a partir de las emisiones de infrarrojos medios del espectro electromagnético de los cuerpos detectados, forma imágenes luminosas visibles por el ojo humano. Todos los objetos emiten energía infrarroja, conocida como señal calórica. Una cámara infrarroja (también conocida como cámara termográfica) detecta y mide la energía infrarroja de los objetos. La cámara convierte los datos infrarrojos en una imagen electrónica que muestra la temperatura aparente de la superficie del objeto medido.

Estas cámaras operan, más concretamente, con longitudes de onda en la zona del infrarrojo térmico, que se considera entre 3 µm y 14 µm (considere el tamaño comparado con las 50 µm que mide el diámetro de un cabello humano). Una cámara térmica o infrarroja permite ver la irradiación de una persona, animal u objeto de lo que

nosotros no podemos ver de lo que llamamos luz visible. Estos dispositivos representan una herramienta muy útil para la detección del índice NDVI en la agricultura, pero su desventaja es el alto costo pues una cámara de este tipo oscila entre los $1,000 USD. En cambio, el prototipo de cámara infrarroja que se presente en este artículo no sobrepasa los $100 USD. En este trabajo presentamos un prototipo de carcasa "funda" para dispositivos móviles equipada con dos lentes de cámara para obtener imágenes en los espectros Infrarrojo y RGB, conectadas a través de una interfaz micro USB (On-The-Go) (ver Fig. 5).

El principal objetivo es capturar imágenes en ambos espectros para poder realizar un procesamiento de pixeles para obtener el índice diferencial normalizado de vegetación que permita determinar el nivel de verdor del objetivo capturado. El prototipo consiste en dos sensores GC0309 con formato óptico de 1/9 de pulgada, se conectan estos sensores a un multiplexor USB clase 1 a la entrada y un cable con conector Micro USB (OTG) a la salida lo cual permite manejar ambos sensores desde el dispositivo móvil evitando así, la conexión y desconexión de cada sensor para obtener ambas imágenes. Se modifica uno de los lentes substituyendo el espejo "caliente" el cual no permite pasar el infrarrojo cercano (NIR – Near Infrared) y sólo deja pasar al sensor el espectro visible de colores (RGB: Red-Green-Blue) por un espejo "frio" que refleja el espectro visible y sólo deja pasar el espectro infrarrojo (NIR) y así obtener las imágenes en dicho espectro. El otro lente queda sin modificaciones para obtener la imagen en el espectro RGB, así obtenemos una cámara multiespectral de bajo costo y altas prestaciones.

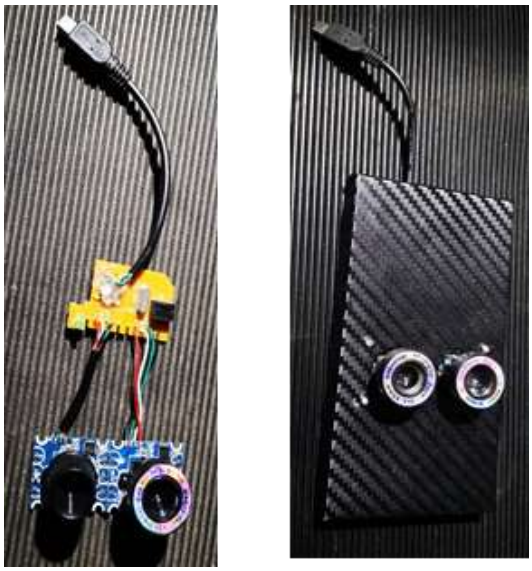


Fig. 5. Prototipo de "carcasa" con doble objetivo.

2.3. Obtención del NDVI

Para obtener el índice NDVI la aplicación NDVICam procesa las imágenes RGB e Infrarroja (ver Fig. 6 superior), leyendo cada valor de pixel rojo de ambas imágenes. El

resultado es una matriz de valores flotantes en el rango de -1 a +1 creando una imagen según la escala de colores conocida como "El rango de valores NDVI" (ver Fig. 6 inferior).

El resultado de esta matriz es representado por un patrón de colores como el que se muestra en la Fig. 8, donde es posible visualizar las zonas con colores "rojizos" que indican que en esa región de la imagen no hay presencia de vegetación; y los colores "amarillentos" representan rangos de valores entre +0.2 a +0.3 (según valores en la gráfica de la Fig. 8, ***esa región de la planta presentaría deficiencias de nutrientes.***

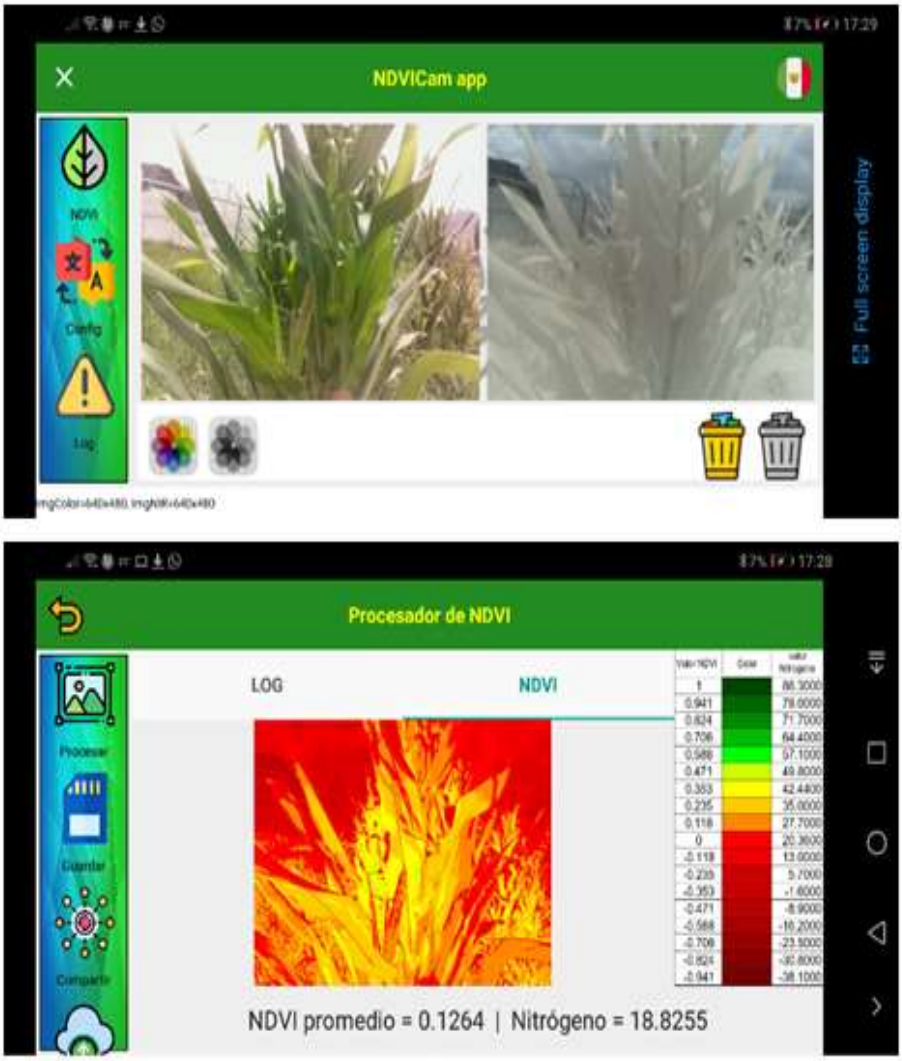


Fig. 6. Elementos del sistema NDVICam.

3. Diseño e implementación

NDVICam está compuesto por los módulos siguientes: a) MOD-IMG o módulo de adquisición de imágenes (trabajo futuro, hasta el momento las imágenes son tomadas mediante el prototipo de carcasa el cual funciona por separado de la aplicación), en esta etapa las imágenes son capturadas por el dispositivo de cámara dual que se ejecuta por separado. Cada imagen es tomada y almacenada en el dispositivo, de esta manera el módulo MOD-IMG carga estas imágenes dentro de la aplicación.

El módulo b) MOD-READ-IMG o módulo para leer las imágenes y obtener los valores correspondientes al color rojo de la imagen RGB y el valor correspondiente al valor NIR (Near Infrared) o infrarrojo cercano que es el mismo valor rojo, pero de la imagen tomada con el espectro infrarrojo, este módulo se basa en la tecnología de hilos (Threads) para aprovechar las capacidades de multiprocesamiento del sistema operativo, donde cada hilo procesa (lee) una imagen por separado.

El módulo c) MOD-NDVI o módulo para obtener el NDVI el cual realiza el cálculo de cada valor NDVI para cada pixel de ambas imágenes, resultando en una matriz de valores flotantes con los rangos obtenidos por la fórmula de NDVI para finalmente presentar el resultado en una escala de colores conocida como "el rango de valores NDVI". La Fig. 7 muestra el diagrama de secuencias del procedimiento para procesar las imágenes y obtener el patrón NDVI.

A continuación, se describe la ejecución de cada módulo hasta la obtención y presentación del índice NDVI.

3.1. MOD-IMG (Módulo de adquisición de imágenes)

Este módulo carga las imágenes capturadas por un dispositivo de cámara dual que se ejecuta por separado; en esta etapa del proyecto el prototipo de cámara con dos lentes instalados, captura dos fotografías del mismo objetivo (esto es porque se requiere obtener el color rojo de ambas imágenes).

Cada imagen es tomada y almacenada en el dispositivo, de esta manera el módulo MOD-IMG carga estas imágenes dentro de la aplicación; este proceso se corresponde con los pasos (2) y (3) del diagrama de secuencias de la Fig. 7. Ambas imágenes deben tener las mismas dimensiones con respecto al ancho y alto de la imagen, de otra forma el usuario recibe un mensaje de advertencia y las imágenes no podrán ser procesadas.

3.2. MOD-READ-IMG (Módulo para leer las bandas de una imagen)

Este módulo realiza un escaneo de cada imagen (pixel por pixel) para obtener el color del canal rojo de cada pixel. Las imágenes están compuestas por 4 canales, cada uno representado por un byte (8 bits) que equivalen a las letras RGBA (R:Red, G: Green, B:Blue y A:Alpha) los cuales indican que 8 bits son dedicados para el color rojo, 8 bits para el verde, 8 bits para el azul (estos componen el color en términos de la intensidad de los tres colores primarios de la luz) y 8 bits para el valor de "Alpha" el cual significa que tan opaco es el pixel.

El módulo rastrea pixel por pixel de la imagen para obtener el nivel de intensidad del color rojo y ser almacenado en un vector, de igual manera se procesa la imagen en el espectro infrarrojo para obtener de la misma manera el color rojo, solo que al ser tomada con un lente que es capaz de captar la luz infrarroja, a este color se le conoce como Infrarrojo cercano o NIR por sus siglas en inglés (Near Infrared).

Para realizar este proceso, el módulo MOD-READ-IMG se basa en la tecnología de hilos (Threads) para aprovechar las capacidades de multiprocesamiento del sistema operativo, donde cada hilo procesa (lee) una imagen por separado y almacena los valores obtenidos en vectores correspondientes al rojo e infrarrojo cercano respectivamente, este proceso se corresponde a los pasos (6a) y (6b) del diagrama de secuencias de la Fig. 7.

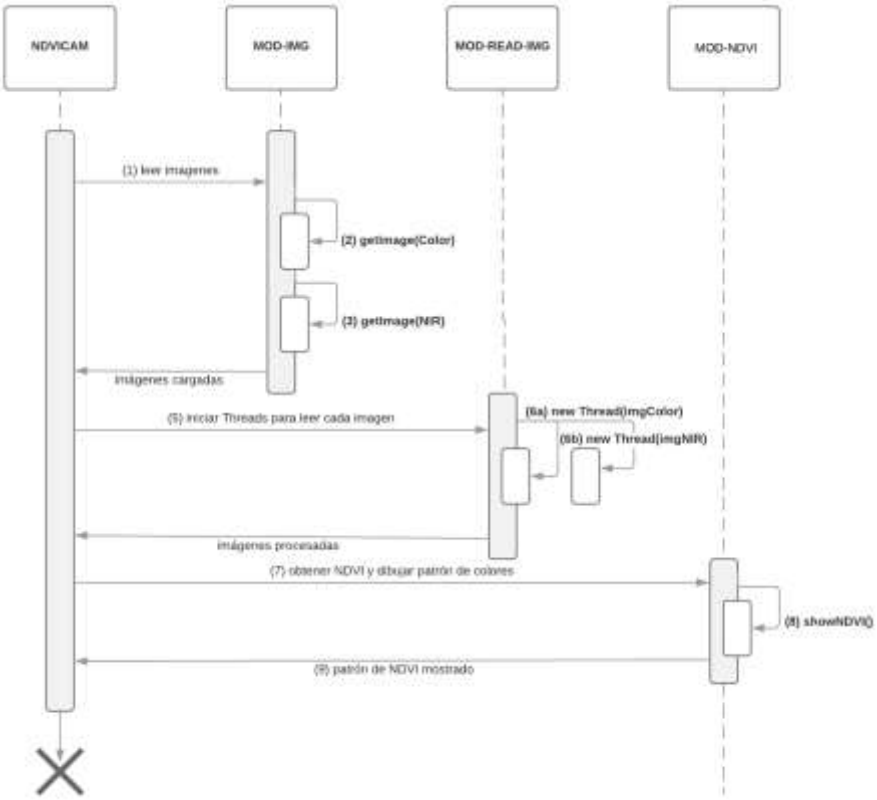


Fig. 7. Diagrama de secuencias para obtener el NDVI.

3.3. MOD-NDVI (módulo para obtener el NDVI)

Este módulo realiza el cálculo de cada valor NDVI para cada valor de pixel rojo de ambas imágenes generado en el módulo MOD-READ-IMG. El resultado es una matriz

de valores flotantes con los rangos obtenidos por la fórmula de NDVI para finalmente presentar el resultado en una escala de colores conocida como "El rango de valores NDVI".

La Fig. 8 muestra el rango de colores para la representación del procesamiento de NDVI. El algoritmo 1, muestra el proceso de las imágenes usando Threads (hilos).

Algoritmo 1. Procesar imágenes usando Threads (hilos).

```
//-----------------------------------------------
// Leer los pixeles de ambas imagenes usando threads
// solo se obtiene el pixel rojo
procedure TfrmNDVI.ReadPixelImages;
begin
  with frmMain do
  begin
    FreeArrays;
    // crear los arreglos según las dimensiones de las imagenes
    //                      rows         cols
    SetLength(arrayRed, imgRedHeight, imgRedWidth);
    SetLength(arrayNIR, imgNIRHeight, imgNIRWidth);
    // ejecutar los threads para leer las imagenes
    frmMain.AddLog('-------------------------');
    TReadImage.Create(false, imgRedFileName, ImageColor, arrayRed);
    TReadImage.Create(false, imgNIRFileName, ImageNIR, arrayNIR);
  end;
end; // ReadPixelImages
```

Este proceso crea dos hilos que se ejecutan en forma paralela aprovechando las capacidades de multiprocesamiento de los dispositivos móviles. Cada thread recibe la referencia a la imagen que ha de procesar (RGB e Infrarroja). El algoritmo 2 muestra como procesa la imagen cada thread.

Algoritmo 2. Código para procesar la imagen dentro del Thread.

```
procedure TReadImage.Execute;
var i, j : integer;
 r: System.Byte;
 image : TBitmapData;
 aColor: TAlphaColor;
begin
  Synchronize(
    procedure
    begin
      FMemo.Lines.Add(Format(aTxt[THREAD_PROCESSING],[FImageFilename,
                      DateTimeToStr(Now)]));
    end);
  // leer el pixel rojo de la imagen
  if FImage.Bitmap.Map(TMapAccess.Read, image) then
  begin
    for i := 0 to Length(FArray)-1 do
    begin
      for j := 0 to Length(FArray[0])-1 do
      begin
```

```
        aColor:= image.GetPixel(i,j);
        r:= TAlphaColorRec(aColor).R;
        FArray[i,j] := r;
      end;
    end;
    FImage.Bitmap.Unmap(image);
  end;
Synchronize(
  procedure
  begin
      FMemo.Lines.Add(Format('<< ' + aTxt[THREAD_PROCESSING],
                     [FImageFilename,
                     DateTimeToStr(Now)]));
  end);
end; // Execute
```

El algoritmo 3 obtiene el valor de la ecuación NDVI para cada pixel de las imágenes procesadas por los threads.

Algoritmo 3. Código para procesar la imagen dentro del Thread.

```
procedure TfrmNDVI.FillNDVIArray;
var i, j : integer;
divisor, dividendo : single;
begin
  // este met. calcula el valor del NDVI leido de ambos arreglos
  // crear el arrgelo NDVI
  SetLength(frmMain.arrayNDVI, frmMain.imgRedWidth,
            frmMain.imgRedHeight);
  Memo1.Lines.Add('****************************************');
  Memo1.Lines.Add(Format('Ini [%s]', [DateTimeToStr(Now)]));
  Memo1.Lines.Add(Format('NDVI[%dx%d]',[Length(frmMain.arrayNDVI),
                  Length(frmMain.arrayNDVI[0])]));
  //con este for se lee hace la formula para cada pixel
  for i := 0 to frmMain.imgRedHeight-1 do
  begin
    for j := 0 to frmMain.imgRedWidth-1 do
    begin
      with frmMain do
      begin
          try
              dividendo := arrayNIR[i,j] - arrayRED[i,j];
              divisor   := arrayNIR[i,j] + arrayRED[i,j];
              arrayNDVI[i,j] := (arrayNIR[i,j] - arrayRED[i,j])
                                /(arrayNIR[i,j] + arrayRED[i,j]);
              //arrayNDVI[i,j] := (dividendo/divisor);
          except on E: Exception do
              // Para la division entre cero
              if divisor = 0
              then arrayNDVI[i,j] := ERROR_DIV_CERO
              else arrayNDVI[i,j] := UNKOWN_NDVI_VALUE;
          end; // try
      end; // with
    end;
  end;
  Memo1.Lines.Add(Format('<< [%s]', [DateTimeToStr(Now)]));
end; // FillNDVIArray
```

3.4. Resultado del procesamiento de las imágenes

NDVICam procesa las imágenes RGB e Infrarroja, para obtener una matriz de valores flotantes con los índices de NDVI correspondientes a cada pixel de la combinación de ambas imágenes. El resultado de esta matriz es representado por un patrón de colores como el que se muestra en la Fig. 8, donde es posible visualizar las zonas con colores "rojizos" que indican que en esa región de la imagen no hay presencia de vegetación; y los colores "amarillentos" representan rangos de valores entre +0.2 a +0.3 (según valores en la gráfica de la Fig. 8) esa región de la planta presentaría deficiencias de nutrientes o algún tipo de anomalía en los nutrientes.

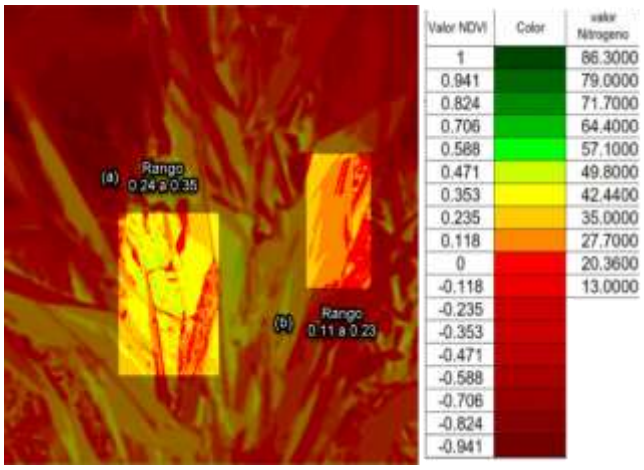


Fig. 8 Relación NDVI con valores de Nitrógeno

Para que el agricultor pueda tener una percepción del estrés en la planta, se realizó una correlación de los valores obtenidos mediante el proceso del NDVI y un dispositivo medidor modelo SPAD-502 (Soil Plant Analysis Development) en plantas de fresa, arándano y aguacate con respecto a la clorofila y el nitrógeno total.

Finalmente, es recomendable medir las unidades NDVI en el cultivo de interés con diferentes hábitos de crecimiento, diferentes estados fenológicos y niveles de nutrición para obtener regresiones que puedan utilizarse en invernadero y campo.

4. Conclusiones

En este artículo se presentó un sistema para el procesamiento de imágenes basado en dispositivos móviles denominado NDVICam. Esta aplicación fue diseñada para obtener el Índice Diferencial de Vegetación Normalizado (NDVI) directamente en los campos de cultivo (in situ) usando dispositivos móviles. NDVICam procesa dos imágenes obteniendo los valores correspondientes al color rojo de la imagen RGB y de la infrarroja conocido como NIR (Near Infrared) o infrarrojo.

Estos valores se procesan para presentar el resultado en una escala de colores correlacionados con el índice NDVI y valores de Nitrógeno. Se presentaron las pruebas realizadas a dos imágenes del mismo objetivo en los espectros de RGB e Infrarrojo respectivamente, para determinar el nivel de salud de la planta con respecto a Nitrógeno.

Asimismo, se presentó la correlación realizada con el dispositivo SPAD el cual está especializado en la medición del Nitrógeno, para poder realizar una correlación con índice NDVI calculado por la aplicación. NDVICam está diseñado para funcionar en Android, iOS, Windows y Mac OSX. Con el proyecto NDVICam los agricultores podrán realizar una detección temprana de factores que afectan el crecimiento o nutrición de los cultivos en cualquier momento y en cualquier lugar.

Agradecimientos. Este trabajo está desarrollado dentro del proyecto de investigación financiado por el Tecnológico Nacional de México (TECNM) titulado "AG-DRONE: Monitoreo de cultivos agrícolas basado en Drones" con clave: 5884.16-P, y desarrollado en el Instituto Tecnológico de Tlajomulco, Jal.

References

1. Aguilar, L.A., Torres-SanMiguel, Ch. R., Martínez, Sáez, L., Lugo León, N., Urriolagoitia Sosa, G., Hernández Gómez, L.H., Urriolagoitia Calderón, G.: Análisis de las propiedades geométricas de las costillas mediante tomografía axial computarizada (TAC) (2014)
2. Bi, X.: LiDAR Technology. In: Environmental Perception Technology for Unmanned Systems. Unmanned System Technologies. Springer, Singapore (2021)
3. Chartuni, M. C.: Manual de agricultura de precisión. Programa Cooperativo para el desarrollo Tecnológico Agroalimentario y Agroindustrial del Cono Sur (PROCISUR), Insituto Interamericano de Cooperación para la Agricultura (IICA) (2014)
4. Chávez, R. O., Clevers, J. G. P. W., Verbesselt, J., Naulin, P. I., Herold, M.: Detecting leaf pulvinar movements on NDVI time series of desert trees: A new approach for water stress detection. PLoS One, 9, 1–12 (2014)
5. Esqueda Elizondo, J., Palafox, L.: Fundamentos de procesamiento de imágenes (2005)
6. Fensholt, R., Sandholt, I., Stisen, S.: Evaluating MODIS, MERIS, and VEGETATION indices using in situ measurements in a semiarid environment. IEEE Trans Geosci, 44, 1774–1786 (2006)
7. Hunt, E. R., Hively, W. D., Fujikawa, S. J., Linden, D. S., Daughtry, C. S. T., McCarty, G. W.: Acquisition of NIR-green-blue digital photos from unmanned aircraft for crop monitoring. Remote Sensing, 2, 290–305 (2010)
8. Jeong, S.: Construction of an unmanned aerial vehicle remote sensing system for crop monitoring. Journal of Applied Remote Sensing 026027-026027 (2016)
9. Martínez, E.: Curso de procesamiento digital de imágenes. Universidad Nacional Autónoma de México, Departamento de ciencias de la computación (2014)
10. Zhang, Xiaoyang: Monitoring vegetation phenology using MODIS. Remote Sensing of Environment 84, 471–475 (2003)

ISSN 1870-4069

Dataset para la detección de elementos de bioseguridad facial mediante técnicas de aprendizaje computacional

Carlos Vicente Niño Rondón, Diego Andrés Castellano Carvajal,
Sergio Alexander Castro Casadiego, Byron Medina Delgado, Dinael Guevara Ibarra

Universidad Francisco de Paula Santander,
Colombia

{carlosvicentenr,diegoandrescc,sergio.castroc,byronmedina,
dinaelgi}@ufps.edu.co

Resumen. En este documento se desarrolla la prueba a un dataset elaborado con tomas de imágenes y videos en la zona céntrica de la ciudad de Cúcuta, con el fin de determinar si las personas que circulan por espacios comerciales portan o no el tapabocas como elemento de bioseguridad facial. El dataset se obtuvo de 1450 imágenes inicialmente capturadas y que, mediante técnicas de aumentado de datos por variaciones en la forma de las imágenes, se aumentaron a 14067 imágenes. Las técnicas de aprendizaje computacional utilizadas son clasificadores en cascada y redes neuronales convolucionales, ambas aplicadas en lenguaje de programación Python. Con el sistema de clasificador en cascada se obtuvo un acierto en las detecciones de 80.17 %, mientras que con la red neuronal convolucional desarrollada se obtuvo un acierto de 90.19 %. Teniendo en cuenta la estructura de las técnicas de aprendizaje, así como las tasas de acierto, se infiere la fiabilidad del conjunto de datos obtenido a la hora de determinar qué personas portan elementos de bioseguridad facial en espacios comerciales.

Palabras clave: dataset, imágenes, clasificador en cascada, red neuronal convolucional.

Dataset for Detection of Facial Biosafety Elements Using Computational Learning Techniques

Abstract. This document develops the test to a dataset elaborated with images and videos taken in the downtown area of the city of Cúcuta, in order to determine whether or not people circulating in commercial spaces wear face masks as an element of facial biosecurity. The dataset was obtained from 1450 images initially captured and, by means of data augmentation techniques due to variations in the shape of the images, was increased to 14067 images. The computational learning techniques used are cascade classifiers and convolutional neural networks, both implemented in Python programming language. With the cascade classifier system, a detection accuracy of 80.17 % was obtained, while with the convolutional neural network developed, an accuracy of 90.19 % was

obtained. Taking into account the structure of the learning techniques, as well as the accuracy rates, the reliability of the data set obtained can be inferred when determining which people are wearing facial biosecurity elements in commercial spaces.

Keywords: dataset, images, cascade classifier, convolutional neural network.

1. Introducción

Los elementos de bioseguridad hacen referencia al conjunto de dispositivos y herramientas utilizados para mitigar las posibilidades de contagio y los efectos generados por diversos agentes biológicos. Durante la pandemia de la COVID-19, los estados y comunidades académicas y científicas han recomendado el uso de elementos de bioseguridad facial, como método para frenar la transmisión del virus entre humanos, que mínimamente se produce con gotas respiratorias de hasta 5 micras en distancias de hasta dos metros [1]. Así mismo, la pandemia trajo consigo estragos en los sectores económicos y productivos de los países, por lo que mediante estrategias conjuntas se ha iniciado con la reactivación económica en los mismos, sin dejar de lado los cuidados mínimos para evitar el aumento acelerado en el número de contagios.

Las técnicas de aprendizaje computacional estudian los algoritmos que generan procesos de aprendizaje mediante guías o series de ejemplos [2]. Además, mediante técnicas de visión por computadora y los métodos de aprendizaje automático o aprendizaje profundo, se adquieren y analizan imágenes del mundo real y se produce información tratable en una computadora [3]. Entre las técnicas de aprendizaje automático para el tratamiento de imágenes destacan los detectores en cascada, donde mediante clasificadores basados en funciones generalmente tipo Haar [4], y en el entrenamiento de imágenes positivas, (correspondientes a imágenes donde se encuentre el objeto en cuestión), e imágenes negativas, (donde no se encuentre el objeto a detectar), se realiza la detección de los mismos, en imágenes que el sistema no ha visto [5]. Por otra parte, entre las técnicas de aprendizaje profundo destacan las redes neuronales convolucionales, donde mediante múltiples capas de filtros de convolución, inicialmente se extraen las características de las imágenes, se reducen por muestreo y se obtienen neuronas simples para ejecutar la clasificación, según las características extraídas [6].

Para aplicar los métodos y procesos de visión y aprendizaje computacional, se requiere de un conjunto de imágenes en las que se presenten las características de el/los objetos a detectar [7]. Dicho conjunto de imágenes o dataset, debe contener un número de imágenes que se encuentran en las escalas de los miles, por lo que, en ocasiones, se requieren de técnicas de aumentado de datos, donde por modificaciones leves de forma en las imágenes originales, se logra la expansión en el número de imágenes para entrenamiento y pruebas [8].

En este documento se presentan las pruebas a un dataset elaborado con imágenes capturadas en espacios comerciales de la ciudad de Cúcuta, Colombia, y completado mediante técnicas de aumentado de datos. El dataset, se prueba mediante un sistema de

clasificadores en cascada y una red neuronal convolucional, ponderando los aciertos en las detecciones en cada uno de los procesos. La adquisición de las imágenes se realiza mediante una cámara de 13 megapíxeles, mientras que la codificación se realiza en lenguaje Python, con herramientas de sistema operativo Windows.

2. Métodos

La metodología propuesta consta de 3 etapas. En la primera etapa se realiza la adquisición de la imagen y se consideran los factores requeridos para la obtención del dataset inicial, como el nivel de luminosidad, hora de la adquisición, altura a la que se ubica el dispositivo de captura de imágenes y videos, y los formatos en los que obtienen las tomas. Asimismo, para la aplicación del proceso de aumentado de datos, se tienen en consideración parámetros de variaciones de forma a las imágenes para el aumentado de datos, como rotación, zoom, desplazamiento de anchura y altura, y rotación horizontal. En la segunda etapa, se aplica el modelo de clasificador en cascada y la red neuronal convolucional, empleando el dataset obtenido inicialmente. En la tercera etapa, se evalúa el rendimiento del dataset mediante las técnicas de aprendizaje computacional empleadas, según los aciertos en las detecciones. La metodología se presenta a continuación, en la figura 1.

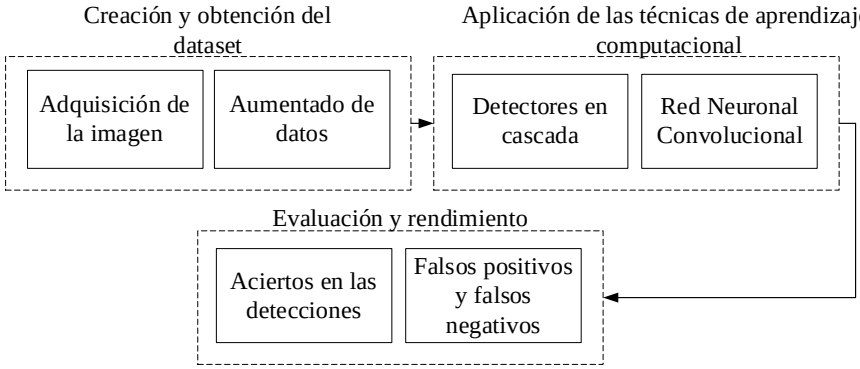


Fig. 1. Metodología propuesta.

2.1. Creación y obtención del dataset

La adquisición de la imagen se realiza en espacios comerciales de la ciudad de Cúcuta, Colombia, mediante una cámara de video de 13 megapíxeles. Las imágenes posteriormente se envían a una computadora personal con procesador Core i7, 4 GB de memoria RAM instalada, sistema operativo de 64 bits con edición Windows 10, donde se realiza el procesamiento. Los elementos de bioseguridad presentes en las imágenes corresponden a tapabocas de tipo higiénico, reutilizables en el caso de los elaborados con tela, así como de un único uso. Además, el dataset también incluye tapabocas de

tipo quirúrgico y de tipo auto-filtrante FFP3. En los tres casos, se incluye una amplia gama de colores, generando así variedad en el conjunto de datos.

Asimismo, en la tabla 1 se presentan los parámetros de caracterización de las zonas en las que se realizó la captura de la imagen. El nivel de luminosidad se obtuvo mediante un luxómetro digital, y estuvo en el rango entre 97 y 4322 luxes, variante entre las 07:00 horas y las 20:00 horas. De igual forma, la altura a la que se ubicó la cámara se encuentra entre 1.7 y 2 metros, obteniendo imágenes en formato JPG y videos en formato MP4.

Tabla 1. Características de la adquisición de la imagen.

Parámetro	Valor
Nivel de luminosidad	[97- 4322] lx
Hora de captura de imagen	[07:00 – 20:00] horas
Altura de localización del dispositivo	[1.70 – 2] m
Formato de imagen/video	JPG/MP4

Para el proceso de aumentado de datos se utilizan los paquetes de Keras y TensorFlow, disponibles para el lenguaje de programación Python [9]. En la tabla 2 se presentan los parámetros requeridos para las modificaciones de forma realizadas sobre las imágenes capturadas inicialmente. Se propone un rango de rotación de hasta 20° para variaciones aleatorias a la imagen. Además, se aplica zoom a la imagen para variar el tamaño aparente de la imagen, en hasta un 17 %. De igual forma, se adaptan variaciones en el desplazamiento de ancho y alto de la imagen con valores de hasta 20 % para cada uno de los casos. Asimismo, se generan imágenes mediante la opción de rotación horizontal, a modo de reflexión en el eje de las X.

Tabla 2. Parámetros de variaciones de forma a las imágenes para el aumentado de datos.

Parámetro	Valor
Angulo de rotación	20°
Zoom	17 %
Desplazamiento de anchura	20 %
Desplazamiento de altura	20 %
Rotación horizontal	Si

2.2. Aplicación de las técnicas de aprendizaje computacional

A continuación, se presentan las estructuras tanto del modelo de clasificador en cascada, como de la red neuronal convolucional.

Clasificador en cascada. Un sistema de clasificadores en cascada se encuentra basado en procesos de aprendizaje automático, donde mediante procesos de entrenamiento, se

evalúa por etapas de modo que se obtenga nueva información sobre los datos de entrada y se genera una función de salida con una clasificación [10]. El modelo de clasificador en cascada se obtiene mediante la aplicación Cascade Trainer GUI, disponible para sistema operativo Windows 7 o superiores [11]. Inicialmente se requieren dos conjuntos de datos correspondientes a imágenes positivas e imágenes negativas. Las imágenes positivas son aquellas en las que se encuentran los tapabocas de tipo higiénico, quirúrgico y auto filtrante, mientras que las imágenes negativas refieren a todos los demás objetos presentes en los ambientes donde se capturó la imagen que no hacen referencia al objeto en cuestión, como vehículos, y prendas de vestir superiores e inferiores; que, para ingresarse a la aplicación, deben alojarse en dos carpetas independientes llamadas "p" para las positivas, y "n" para las negativas.

La aplicación Cascade Trainer GUI en su ventana principal presenta 4 pestañas llamadas Input, Common, Cascade, y Boost, y en cada una de las mismas se realiza la configuración para la obtención del modelo de clasificador. En Input se realiza el cargue de las imágenes positivas y negativas, y se define el porcentaje de imágenes positivas a utilizar para el entrenamiento, que para el presenta caso es del 100 %. Asimismo, en la pestaña Common, se configura entra otros, el número de etapas e hilos, siendo 10 y 5 dichos valores respectivamente. En la pestaña Cascade se configuran los valores de ancho y alto de las muestras, siendo imágenes de tamaño 24x24, además del tipo de caracteristicas, siendo en el presente caso de tipo HAAR, y, en la pestaña Boost, se define el tipo de impulso para el modelo de clasificación, siendo GAB el definido, basado en el Algoritmo Gentle Adaboost, además de la tasa mínima de acierto, definida en 0.995. En la figura 2 se presenta la estructura del modelo de clasificador en cascada, donde se ilustra el cargue de las imágenes a la aplicación Cascade Trainer GUI, y la obtención del archivo .XML que contiene el modelo de clasificación.

Fig. 2. Estructura para la obtención del modelo de clasificador en cascada.

Red neuronal convolucional. Las redes neuronales convolucionales (CNN) son un tipo de red neuronal artificial que se encuentra basada procesos de aprendizaje profundo. El aprendizaje de la red se basa en captar las características únicas de los objetos y generalizarlos [12]. La estructura básica de una CNN consta de 5 etapas denominadas entrada, convolución, activación, muestreo, y obtención de probabilidades [13]. La entrada hace referencia a los pixeles de imagen y características de tamaño y color. En las capas de convolución se procesan las neuronas conectadas mediante pixeles vecinos, y según la cantidad de filtros se determina el volumen de salida. Las funciones de activación refieren a los métodos para propagar la salida de los nodos de una capa hacia las demás capas de la red. En la etapa de muestreo, se realizan las modificaciones de ancho y alto de los datos, sin alterar la profundidad de la

información en proceso. Finalmente, en la etapa de obtención de probabilidades, la red arroja su predicción respecto a la categoría a la que pertenece el objeto presente en la imagen ingresada al sistema [14].

Para el desarrollo de la red neuronal convolucional en lenguaje Python, se requiere de los paquetes de Keras y TensorFlow. En la tabla 3 se presentan los parámetros de configuración de la red neuronal. Se proponen 20 épocas para el entrenamiento de la red, con pasos de 32 imágenes de tamaño 200x200 pixeles. Además, se ejecutan 2 filtros de convolución con tamaños de 2x2 y 3x3 respectivamente. Asimismo, para la reducción en las dimensiones de los datos se utiliza la función MaxPooling, que toma el mayor valor al ejecutarse mediante un filtro de 2x2 [15]. La función de activación utilizada es la de tipo ReLU, y adicionalmente, mediante el Dropout de desactivan instantáneamente el 50 % de las neuronas, evitando así un único camino de entrenamiento. De igual forma, para estimar los valores de las predicciones de salida (persona con tapabocas o persona sin tapabocas) se utiliza la función Softmax.

Tabla 3. Parámetros de configuración de la red neuronal convolucional.

Parámetro	Valor
Épocas	20
Pasos	32
Dimensiones de entrada	200x200
Cantidad de filtros de convolución	2
Tamaño de filtros de convolución	2x2 y 3x3
Tamaño y estructura del Pooling	2x2, MaxPooling
Función de activación	ReLU
Dropout	0.5
Función de probabilidades	Softmax

2.3. Evaluación y rendimiento

Las pruebas al dataset se realizan mediante la medición de falsos positivos y negativos, así como la ponderación de verdaderos positivos y verdaderos negativos en el caso del sistema de clasificador en cascada, y al igual que para la red neuronal convolucional mediante una matriz de confusión se ponderan los de aciertos en las predicciones. Para cada uno de los casos, se calcula el error medio y de esta forma se determina el porcentaje de inconsistencias en las clasificaciones.

3. Resultados

En la tabla 4 se presenta la relación entre el número de imágenes capturadas inicialmente y las obtenidas posterior al aplicar las técnicas de aumentado de datos. Un total de 1,450 imágenes inicialmente se aumentan en aproximadamente 10 veces,

obteniendo 14,067 imágenes, distribuidas en 7,174 imágenes de personas con tapabocas y 6,893 personas sin tapabocas. La eficiencia con la que se realizó el aumentado de las imágenes fue de 89.73 % y 88.81 % respectivamente.

Tabla 4. Relación entre el número de imágenes capturadas y obtenidas posterior al aumentado de datos.

Categoría	Antes	Después	Eficiencia en el aumentado
Con tapabocas	738	7174	89.73%
Sin tapabocas	712	6893	88.81%
Total	1450	14067	

De igual forma, el tiempo empleado para el entrenamiento del sistema de clasificadores en cascada fue de 83 minutos, mientras que la red neuronal convolucional requirió de 247 minutos en entrenarse. El tiempo utilizado en el entrenamiento es dependiente tanto de la estructura de los modelos, como de la herramienta de hardware utilizada en el procesamiento, que, en el presente caso, es una computadora personal con procesador Core i7, 4 GB de memoria RAM instalada y sistema operativo Windows 10.

Así mismo, en la figura 3 se presentan los resultados obtenidos en las predicciones de las técnicas de aprendizaje computacional. Para el modelo de clasificador en cascada, de 73 imágenes de personas sin tapabocas, se logró la estimación correcta en 67 de las mismas, mientras que en 6 oportunidades los clasificó como si portaran el tapabocas. Por otra parte, de 72 personas que portaban el tapabocas, se estimó correctamente la predicción de 59 de ellos, mientras que en 13 oportunidades se categorizó como si las personas no portaran el tapabocas. Asimismo, al aplicar la red neuronal convolucional con el mismo conjunto de imágenes de prueba, se logró la predicción acertada de 72 personas que no portaban el tapabocas, y de 65 personas que si lo portaban.

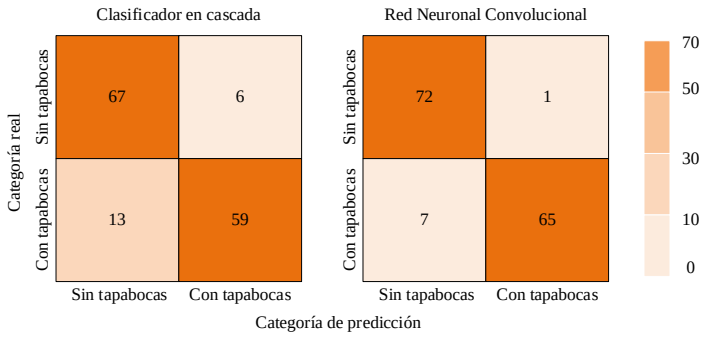


Fig. 3. Matriz de confusión para las técnicas de aprendizaje computacional.

Así mismo, en las ecuaciones 1 y 2 se muestra la forma en cómo se obtuvo el error medio en las predicciones al aplicar el modelo de clasificador en cascada y la red

neuronal convolucional respectivamente. En el sistema de clasificadores en cascada se genera un error de 19.83 %, mientras que, para la red neuronal convolucional, el error fue de 9.81 %:

$$\sqrt{\left(\frac{6}{73}\right)^2 + \left(\frac{13}{72}\right)^2} = 0.1983 \; o \; 19.83\%, \tag{1}$$

$$\sqrt{\left(\frac{1}{73}\right)^2 + \left(\frac{7}{72}\right)^2} = 0.981 \; o \; 9.81\%. \tag{2}$$

4. Conclusiones

Las estructuras expuestas, tanto para el método de aprendizaje automático como para el de aprendizaje profundo, pueden definirse como estructuras comunes en los modelos de aprendizaje computacional, y emulan un comportamiento preliminar del conjunto de datos ante este tipo de entrenamiento, por lo que, mejorando y potenciando las fases de entrenamiento, se potencializa el comportamiento del dataset.

Dicho dataset al ser elaborado con imágenes de contextos y ambientes reales, con niveles de luminosidad variantes a lo largo del día y con tomas desde alturas donde se logra percibir los elementos de bioseguridad facial utilizados por las personas, y probados en espacios comerciales de la ciudad de Cúcuta, presenta una robustez en el comportamiento de las técnicas de aprendizaje computacional utilizadas, ya que logran aciertos en la clasificación superiores al 80 %.

Asimismo, la red neuronal convolucional mejora en aproximadamente un 10 % el comportamiento en las clasificaciones respecto al modelo de clasificador en cascada, generado por el número de etapas de entrenamiento y procesos computacionales propios de los procesos de aprendizaje profundo.

Referencias

1. Shereen, M.A., Khan, S., Kazmi, A., Bashir, N., Siddique, R.: COVID-19 infection: Origin, transmission, and characteristics of human coronaviruses (2020). https://doi.org/10.1016/j.jare.2020.03.005.
2. Zou, X.: A Review of object detection techniques. In: Proc. Int. Conf. Smart Grid Electr. Autom. ICSGEA, pp. 251–254 (2019). https://doi.org/10.1109/ICSGEA.2019.00065.
3. Ramírez Jiménez, D., Quintero Ospina, J.D.: Clasificación de patologías presentes en la columna vertebral mediante técnicas de máquinas de aprendizaje. Ing. Solidar. 15, 1–23 (2019). https://doi.org/10.16925/2357-6014.2019.01.05.
4. Selvaraj, K., Fathima, A.A., Vaidehi, V.: Multi-class object detection by part based approach. In: International Conference on Recent Trends in Information Technology, ICRTIT 2012. pp. 114–118 (2012). https://doi.org/10.1109/ICRTIT.2012.6206837.
5. Niño Rondón, C.V., Castro Casadiego, S.A., Medina Delgado, B., Guevara Ibarra, D., Ramirez Mateus, J.J., Puerto López, K.C.: Comparación multiplaforma de técnicas basadas en visión artificial para detección de personas en espacios abiertos. Investig. e Innovación en Ing. 9, 22–33 (2021). https://doi.org/10.17081/invinno.9.1.3965.

6. Shorten, C., Khoshgoftaar, T.M.: A survey on Image Data Augmentation for Deep Learning. J. Big Data. 6 (2019). https://doi.org/10.1186/s40537-019-0197-0.
7. Plebani, E., Celona, L., Pau, D., Karimi, P., Marcon, M.: Training an object detector using only positive samples. In: 2015 IEEE 1st International Workshop on Consumer Electronics - Novi Sad, CE WS 2015. pp. 1–4. Institute of Electrical and Electronics Engineers Inc. (2017). https://doi.org/10.1109/CEWS.2015.7867139.
8. Mikołajczyk, A., Grochowski, M.: Data augmentation for improving deep learning in image classification problem. In: 2018 International Interdisciplinary PhD Workshop, IIPhDW 2018. pp. 117–122. Institute of Electrical and Electronics Engineers Inc. (2018). https://doi.org/10.1109/IIPHDW.2018.8388338.
9. Kakuda, K., Enomoto, T., Miura, S.: Nonlinear activation functions in CNN based on fluid dynamics and its applications. C. - Comput. Model. Eng. Sci. 118, 1–14 (2019). https://doi.org/10.31614/cmes.2019.04676.
10. Siqueira, D.L., Manso Correa Machado, A.: People Detection and Tracking in Low Frame-rate Dynamic Scenes. IEEE Lat. Am. Trans. 14, 1966–1971 (2016). https://doi.org/10.1109/TLA.2016.7483541.
11. Phase, T.R., S. Patil, S.: Building Custom HAAR-Cascade Classifier for face Detection. Int. J. Eng. Res. V. 8 (2020). https://doi.org/10.17577/ijertv8is120350.
12. Öztürk, Ş., Akdemir, B.: Effects of Histopathological Image Pre-processing on Convolutional Neural Networks. In: Procedia Computer Science. pp. 396–403. Elsevier B.V. (2018). https://doi.org/10.1016/j.procs.2018.05.166.
13. Krishna Sai, B.N., Sasikala, T.: Object Detection and Count of Objects in Image using Tensor Flow Object Detection API. In: Proceedings of the 2nd International Conference on Smart Systems and Inventive Technology, ICSSIT 2019. pp. 542–546. Institute of Electrical and Electronics Engineers Inc. (2019). https://doi.org/10.1109/ICSSIT46314.2019.8987942.
14. Agarap, A.F.: Deep Learning using Rectified Linear Units (ReLU). arXiv. (2018).
15. Suárez Paniagua, V., Segura Bedmar, I.: Evaluation of pooling operations in convolutional architectures for drug-drug interaction extraction. BMC Bioinformatics. 19, 209 (2018). https://doi.org/10.1186/s12859-018-2195-1.

ISSN 1870-4069

Creación de planes alimenticios mediante algoritmos genéticos para combatir la obesidad infantil en México

Cristian I. Echartea de la Rosa, Marco Aurelio Nuño Maganda, Yahir Hernández Mier, Said Polanco Martagón

Universidad Politécnica de Victoria, Victoria, Tamaulipas, México

{1530229, mnunom, yhernandezm, spolancom}@upv.edu.mx

Resumen. En la actualidad (2021), la obesidad infantil es un problema grave de salud que va en aumento aunado de las enfermedades que se pueden producir o agravar a causa de esta, prueba de ello se ve reflejado en el gran impacto que ha tenido el padecer esta enfermedad ante el COVID-19, es por ello que se deben de crear estrategias para combatirla. El objetivo de este artículo es la creación de un Algoritmo Genético el cual genere un plan alimenticio adecuado a las características de cada niño con el fin de combatir la obesidad en México. El plan alimenticio será conformado por distintos alimentos que se encuentran en una base de datos, el contenido calórico de cada alimento fue asignado por un nutriólogo. El propósito del algoritmo es acercar el IMC del niño a su percentil ideal. Los resultados muestran que a pesar de que la mayoría de los alimentos tienen un alto contenido calórico el algoritmo obtuvo buenas soluciones, se encontró una correlación con el periodo y la actividad física que se realiza. Se concluyó que los algoritmos genéticos son una buena alternativa para la creación de planes alimenticios, este enfoque podría ser utilizado por profesionales y la población en general.

Palabras claves: Plan alimenticio, salud, sobrepeso, computación evolutiva.

Creation of Eating Plans Using Genetic Algorithms for Combating Childhood Obesity in Mexico

Abstract. Nowadays (2021), childhood obesity is a serious health problem that is increasing, in addition to the diseases that can occur or aggravate for that, proof of this is reflected in the great impact that the disease has had on COVID-19, that is why strategies must be created to combat it. The objective of this article is create a Genetic Algorithm which generates a food plan appropriate to the characteristics of each child in order to combat obesity in Mexico. The food plan will be made

up of different foods found in a database, the caloric content of each food was assigned by a nutritionist. The main idea of the algorithm is to bring the child's BMI closer to its ideal percentile. The results show that although most foods have biggest calories content the algorithm obtained good solutions, a correlation was found with the period and physical activity that is performed. It was concluded that genetic algorithms are a good alternative for the creation of food plans, this approach could be used by professionals and the general population.

Keywords: Eating plan, health, overweight, evolutionary computation.

1. Introducción

Hoy en día la obesidad infantil es un problema grave que afecta tanto a países de bajo y alto nivel económico, es considerada como una enfermedad desde 1955. Se define como un padecimiento crónico complejo de etiología multifactorial que se desarrolla por un desequilibrio entre la energía ingerida y gastada [2]. Este padecimiento es provocado principalmente por factores genéticos, psicológicos, socioeconómicos y el llevar un estilo de vida sedentario. Un método de diagnóstico, es por el Índice de Masa Corporal (IMC), se calcula $\frac{Peso_{kg}}{Estatura_{mts^2}}$.

Para personas mayores de 20 años si el resultado es menor a 18.5 quiere decir que tienen infra peso, de 18. 5 a 24.9 peso normal, de 25.0 a 29.9 sobrepeso y de 30 en adelante obesidad, en el caso pediátrico hay una gráfica de percentiles de acuerdo a la edad.

La Organización Mundial de la Salud (OMS) informó que cada año mueren como mínimo 2,8 millones de personas a causa de este padecimiento [12]. La Organización Panamericana de la Salud (OPS) manifiesta que el número de niños de 5 a 19 años de edad que la presentan, se ha decuplicado en las cuatro últimas décadas [13]. En cuanto a México cifras del Instituto Nacional de Estadística y Geografía (INEGI) referentes a la población infantil, mostraron un incremento en los casos de un 2.6 % del 2012 al 2018 [8].

Desafortunadamente a causa de esta enfermedad pueden producirse o agravarse otras enfermedades como la diabetes, síndrome metabólico, tumores, enfermedades cardiovasculares y COVID-19. Además, se ven implicados factores sociales y psicológicos [6,9,14,3,5].

Investigaciones previas muestran que la Inteligencia Artificial (IA) se encuentra presente en esta lucha. López y Zamora en 2018 propusieron una aplicación móvil que genere dietas por medio de un Algoritmo Genético (AG) con base al requerimiento calórico [10]. Por otra parte, Catalán et al. [4], proponen un AG para generar dietas que reduzcan el índice de sobrepeso u obesidad. De igual forma Wulandhari et al., crearon dos algoritmos de optimización para encontrar una buena combinación y porción de alimentos [20]. Por último, el siguiente artículo propone comparar 4 algoritmos aplicados a la optimización de la dieta en general [1]. Todas las investigaciones tenían como propósito mejorar

los hábitos alimenticios, concluyeron que los algoritmos de optimización son una buena alternativa para generar un Plan Alimenticio (PA).

Tomando en cuenta la gran cantidad de enfermedades que se producen o agravan a causa de la obesidad y la gran problemática que se está viviendo actualmente por el COVID-19, se deben proponer estrategias en el combate en contra de ella. Es por eso que el objetivo de este artículo es la creación de un AG el cual genere un PA adecuado a las características del usuario con el fin de combatir la obesidad infantil en México.

Las bases del AG fueron asentadas desde 1962 por John Holland, es un método de búsqueda que imita la teoría de la evolución biológica de Darwin para la resolución de problemas [15]. Los AG son métodos adaptativos, generalmente usados en problemas de optimización de parámetros, basados en la reproducción y en el principio de supervivencia del más apto [16].

El GA servirá para seleccionar un PA, el cual contendrá 3 comidas elegidas de una base de datos compuesta de alimentos comunes en México, este tomará en cuenta la Actividad Física (AF) con el fin de encontrar una buena alternativa que acerque el IMC al percentil ideal de acuerdo a estándares de la OMS [12].

En la primera fase se contempla solamente el almuerzo, comida y cena, debido a que la mayoría de las personas en México solo ingieren 3 comidas por día [17]. Los alimentos fueron seleccionados de las opiniones de un grupo de 9 personas, la lista consta de más de 140 alimentos con alto contenido calórico, con los cuales el AG trabajará.

Para el funcionamiento, crecimiento y desarrollo del cuerpo, el ser humano debe de ingerir nutrientes que aporten la energía necesaria al organismo. Las necesidades energéticas de un humano varían entre 1.000 y 4.000 Kcal/día dependiendo de sus características [18]. Las calorías son la energía que recibe el cuerpo por medio de los alimentos.

Para obtener el Gasto de Energía Total (GET) como se muestra en la ecuación 1, se necesita el Gasto de Energía del cuerpo estando en Reposo (GER), la cantidad de AF que realiza la persona y el Efecto Térmico de los Alimentos (ETA) que es aproximadamente el 10 % del GER [7]. Hay diversas ponderaciones que se le da a la AF dependiendo de la cantidad que se realice [19]. Se recomienda la ecuación de Schofield para niños y adultos [11]. Las fórmulas se muestran en la tabla 1:

$$GET = (GER \times AF) + ETA. \tag{1}$$

Tabla 1. Ecuación de Schofield. P: Peso (KG) E; Estatura (CM).

Sexo	**Edad (Años)**	**Ecuación**
Hombres	3-10	$(19.59 \times P) + (1{,}303 \times E) + 414{,}9$
	10-18	$(16.25 \times P) + (1{,}372 \times E) + 515{,}5$
Mujeres	3-10	$(16.969 \times P) + (1{,}618 \times E) + 371{,}2$
	10-18	$(8.365 \times P) + (4{,}65 \times E) + 200$

2. Métodos

La problemática se resolverá por medio de un AG, este algoritmo comienza por la creación de un conjunto de soluciones llamada Población Inicial (PI), se selecciona a las mejores soluciones, a partir estás se hace la Cruza con base a una Probabilidad de Cruza (PC), para crear nuevos individuos hijos que contendrán información de los padres (soluciones) más aptos de la generación, estos podrían ser multados de acuerdo a una Probabilidad de Mutación (PM).

Una vez que se tiene a los padres e hijos se hará el elitismo, consiste en eliminar a los más débiles, los más aptos serán los padres o PI de la siguiente generación. Este proceso se hará por un Número de Generaciones (NG) tratando de encontrar una solución factible como se muesta en el Algoritmo 1.

Algorithm 1 Pseudocodigo del AG.

```
Generar población inicial = N
for i=0 : Generaciones do
    Selección
    Cruza
    Mutación
    Elitismo
end for
```

El individuo será representado por una matriz de 3 x 7 como se muestra en la Figura 1.

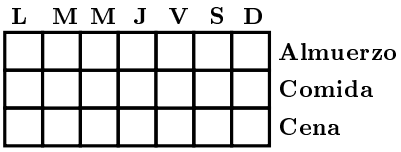


Fig. 1. Individuo o solución con la que trabaja el AG.

Son necesarias una serie de ecuaciones para el funcionamiento del AG, en la ecuación 2, se obtiene la Diferencia Calórica (DC) del GET simulado (GETS) y el GET necesario en el periodo de simulación (GETT).

En la ecuación 3, se convierte la DC en kg, siendo el aumento o decremento de peso(WS) en kg, en la ecuación 4, (W) es el peso actual al que se le agregara WS con el fin de saber si aumenta o decrementa por lo tanto W' es el peso simulado.

En la ecuación 5, el Índice de Masa Corporal Simulado (IMCS), se calcula en base al W' dividido entre la altura. Finalmente, el ADP está dado por la ecuación 6, y se estima en base a la diferencia absoluta del IMC ideal (IMCI)

y IMCS:

$$DC = (GET \times 7 \times semanas)(GETS \times semanas), \tag{2}$$

$$WS = DC/7000, \tag{3}$$

$$W' = W + WS, \tag{4}$$

$$IMCS = \frac{W'}{Altura_{mts^2}}, \tag{5}$$

$$Adp = abs(IMCSIMCI). \tag{6}$$

Como parte del desarrollo, los tipos de selección que se utilizaron fueron Selección Ruleta (SR) y Selección Jerárquica (SJ), en la SR cada uno de los individuos de la población se le asigna una porción proporcional a su adaptación y en forma de lista se va sumando la proporción actual más la del individuo anterior hasta alcanzar la unidad simulando una ruleta, posteriormente se genera un número entre 0 y 1, el individuo que tenga la proporción más cercana será el elegido.

La SJ es semejante a la ruleta solo que antes de sacar la proporción de cada uno los individuos deben de ordenarse de acuerdo a la adaptación con el fin de que aumente la posibilidad de ser seleccionado al tener una mejor adaptación. Mientras que en la cruza se utilizó Cruza Monopunto y Cruza Multipunto.

En la cruza Monopunto se generó un Punto de Cruza (PC) de 1 a N, el padre 1 aportará la información desde la primera columna hasta PC y el padre dos desde PC hasta la última columna figura 2.

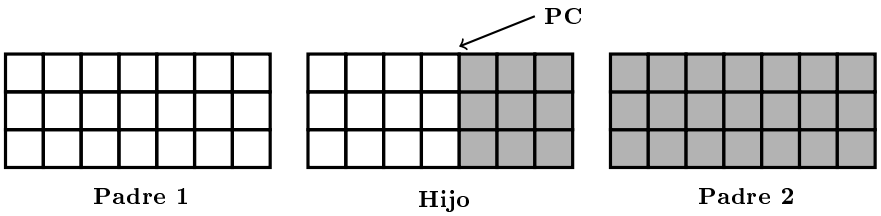


Fig. 2. Cruza monopunto del AG.

En la cruza multipunto el padre 1 y 2 tienen la misma probabilidad de aportar información al hijo, se genera un número aleatorio de 0.0 a 1.0, si el número era menor a 0.5 se le asignaba la información del padre 1 de lo contrario se le asignaba del padre 2 figura 3. Se optó por tener dos tipos de cruza con el fin de comparar qué método arroja mejores resultados.

3. Resultados

Para el funcionamiento, el GA necesita parámetros como la Edad y Meses, Peso(kg), Estatura(cm), Sexo, AF y Numero de Semanas (NS). La AF va del 1

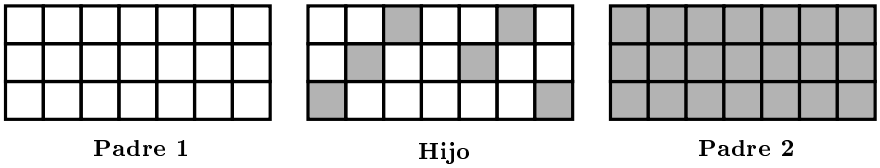


Fig. 3. Cruza multipunto del AG.

a 5 de acuerdo a la intensidad, mientras el NS representa el periodo del PA. Las ejecuciones (Ej) se muestran en la tabla 2.

Tabla 2. Pruebas de ejecución del AG.

Ej	Edad	Peso	Sex	Est	AF	NS	W'	IMCI	IMCS	GETT	GETS	NG
1	11,9	44.85	H	159	3	6	42.59	16.85	16.85	103997.00	88242.0	100
2	11,9	44.85	H	159	5	6	41.11	16.85	16.26	126057.00	99909.0	100
3	11,9	44.85	M	159	3	6	43.48	17.20	17.19	96293.85	86724.0	100
4	11,9	34.85	M	159	1	4	40.18	17.20	15.80	44401.87	81780.0	100
5	10,5	37.00	H	151	2	7	36.70	16.10	16.10	96594.30	94563.0	100
6	12,2	59.00	H	162	1	8	45.13	17.20	17.19	129714.70	32692.0	100
7	12,7	63.00	M	163	1	8	47.82	18.00	17.99	124049.80	17816.0	100
8	11,9	55.00	M	159	1	6	45.19	17.20	17.19	85271.93	9318.0	130
9	11,9	34.00	H	159	1	6	41.69	16.85	16.49	70332.28	124182.0	130

En la Ej. 1 se observa que, aunque el peso decrementa en la simulación el niño se mantiene muy cerca del promedio del IMC ideal. La Ej. 2 contiene los mismos parámetros de la Ej. 1 a diferencia que aumentó al máximo la AF, el resultado mostró que el GETS aumentó a comparación de la Ej. 1 ya que al hacer más AF el cuerpo requiere más energía. La Ej. 3 contiene los mismos parámetros que la Ej. 1 para un sexo distinto, las niñas tienen un rango más alto de IMC y el GETT es menor que al de un niño de su edad, es por ello que se refleja una diferencia en los resultados, para todos los casos la PM y PC fueron de 0.3 mientras que la PI fue de 100, los tipos de selección y cruza que arrojaron mejores resultados fueron SJ y cruza multipunto.

De manera más ilustrativa se muestra la representación de la Ej. 8 en la figura 4, se observa cómo se va acercando el IMCS al IMCI, para este caso el IMCS tenderá a disminuir debido a que el infante está por arriba de su peso ideal, el PA seleccionado para esta Ej. se muestra en la tabla 3, el PA es muy bajo en calorías debido a que el infante tiene que disminuir su peso y su AF es nula.

De igual forma para la Ej. 9 su representación se muestra en la figura 5, en la gráfica se observa como el IMCS sube, esto debió a que el IMC inicial del niño está por debajo del normal, el AG buscará igualar el IMCS al IMCI. El plan alimenticio para esta ejecución se muestra en la tabla 4, este plan es alto

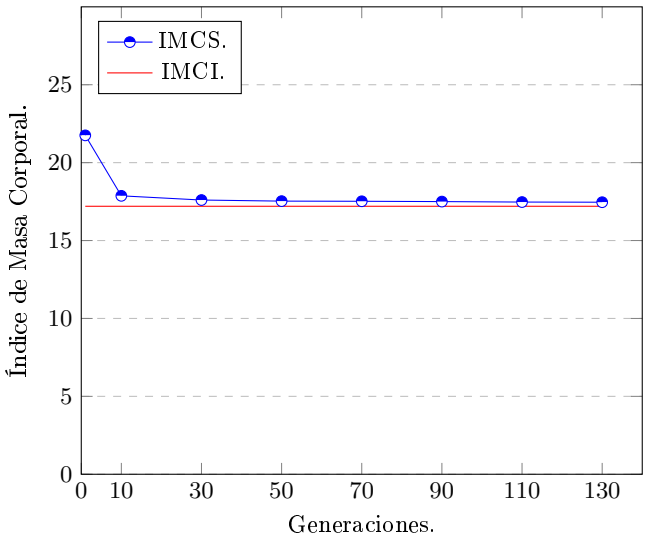


Fig. 4. Evolución del IMCS a lo largo de las generaciones del AG, Ej. 8.

Tabla 3. PA generado por el AG en la Ej. 8.

	Lunes	**Martes**	**Miércoles**	**Jueves**	**Viernes**	**Sábado**	**Domingo**
Almuerzo	1 plátano **60.0 cal**	1 taza de café **4.0 cal**	1 huevo estrellado **69.0 cal**	1 Fruta **60.0 cal**	taza de café **4.0 cal**	1 Fruta **60.0 cal**	1 Fruta **60.0 cal**
Comida	1 chile relleno de picadillo **84.0 cal**	Un plato de ensalada **25.0 cal**	Pollo frito **106.0 cal**	Sopa de fideo con pollo **118.0 cal**	1 milanesa de res con arroz **96.0 cal**	Espagueti con milanesa de pollo **89.0 cal**	1 chile relleno de picadillo **84.0 cal**
Cena	1 Huevo con chorizo **127.0 cal**	1 huevo con tomate **88.0 cal**	1 Huevo en torta **63.0 cal**	1 vaso de Avena **77.0 cal**	1 vaso de Avena **77.0 cal**	Ensalada de pollo **139.0** cal	1 Huevo en torta **63.0 cal**

en calorías debido a que el infante tiene que aumentar de peso aunado de que su AF es nula.

En cuanto al AG los tiempos de ejecución fueron cortos ya que se probó y selecciono un número de PI y generaciones bajo que diera buenas alternativas. Se obtuvieron buenos resultados en cuanto al PA, la simulación muestra que en todos los casos el IMCS del infante se acerca a su IMCI. Se pudo observar una relación entre el NS y la AF, aunque desde nuestro punto de vista, el valor de NS no debe de ser superior a 4 ya que parámetros como estatura y peso eventualmente cambiarán.

Profesionales médicos recomiendan alternar el PA con la AF y llevar un PA que paulatinamente vaya subiendo o disminuyendo el contenido calórico de tal manera que no afecte la salud de la persona al tener un cambio abismal en su ingesta. De igual forma se deben tomar en cuenta restricciones con personas que tienen problemas de salud ya que estas no pueden llevar a cabo un PA sin que

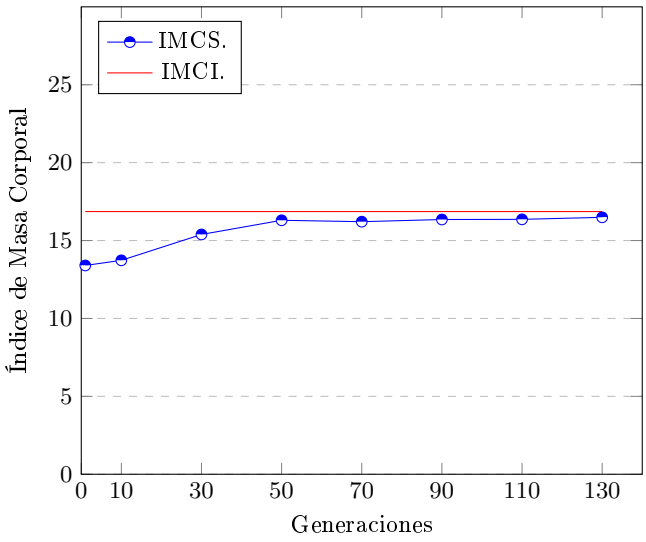


Fig. 5. Evolución del IMCS a lo largo de las generaciones del AG, Ej. 9.

Tabla 4. PA generado por el AG en la Ej. 9.

	Lunes	**Martes**	**Miércoles**	**Jueves**	**Viernes**	**Sábado**	**Domingo**
Almuerzo	3 gorditas de salsa verde **1041.0 cal**	3 gorditas de chicharron **996.0 cal**	3 gorditas de salsa verde **1041.0 cal**	3 gorditas de salsa verde **1041.0 cal**	3 gorditas de salsa verde **1041.0 cal**	3 gorditas de salsa verde **1041.0 cal**	3 gorditas de salsa verde **1041.0 cal**
Comida	4 flautas de salsa verde **602.0 cal**	2 rebanadas de pizza **1020.0 cal**	2 rebanadas de pizza **1020.0 cal**	2 rebanadas de pizza **1020.0 cal**	2 rebanadas de pizza **1020.0 cal**	2 rebanadas de pizza **1020.0 cal**	2 rebanadas de pizza **1020.0 cal**
Cena	2 rebanadas de Pizza **1020.0 cal**	2 rebanadas de Pizza **1020.0 cal**	4 tostadas de frijoles con queso **944.0 cal**	3 rebanadas de Pizza **765.0 cal**	2 rebanadas de Pizza **1020.0 cal**	2 rebanadas de Pizza **1020.0 cal**	4 tostadas de frijoles con queso **944.0 cal**

este haya sido autorizado por un especialista, es por ello que los PA no han sido aplicados en personas, pero los resultados obtenidos son muy prometedores.

A diferencia de otras investigaciones, esta está enfocada en la población pediátrica tomando como referencia el percentil del IMC de la OMS, los cuales se encuentran regulados. Además, el GA simula el aumento de peso en un periodo determinado de acuerdo a la carga calórica del PA, mientras que los algoritmos similares solo trabajaban encontrando un PA de acuerdo a las calorías.

4. Conclusiones

En esta investigación se planteó la aplicación de un AG orientado a la generación de PA como una herramienta para combatir la obesidad infantil. El trabajo

concluye con la implementación de un AG para la creación de PA, en cuanto a su validación se hicieron pruebas simuladas sin que alguna persona llevará a cabo el PA.

Los resultados obtenidos muestran que, en todas las pruebas realizadas, el IMCS del niño(a) se acercó al IMCI, se observó una correlación entre el tipo de AF y el NS del PA, se observó que el contenido calórico del PA es muy similar al que se obtuvo con la ecuación predictiva, por ello y por las altas expectativas que generan los resultados se concluye que los AG son una buena alternativa para crear un PA. Esta alternativa podría ser utilizada en niños para acercarlos o mantenerlos en su IMC ideal para combatir la obesidad, siempre estando bajo la supervisión de un profesional en la salud.

Agradecimientos Agradezco al CONACYT por la beca (CVU 1014356) que me otorgó para la realización de mis estudios de posgrado, del cual se derivaron los resultados alcanzados en este proyecto.

Referencias

1. Babalola, A.E., Ojokoh, B.A., Odili, J.B.: Diet optimization techniques: A review. In: 2020 International Conference in Mathematics, Computer Engineering and Computer Science (ICMCECS). pp. 1–5. IEEE (2020)
2. Blancas-Flores, G., Almanza-Pérez, J.C., López-Roa, R.I., Alarcón-Aguilar, F.J., García-Macedo, R., Cruz, M.: La obesidad como un proceso inflamatorio. Boletín médico del Hospital Infantil de México 67(2), 88–97 (2010)
3. Carnero, M.G., Álvarez, P.F., Molares, A.V., Álvarez, M.G., Carnero, O.G., Arias, J.Á., Blach, M.I., Villaverde, C.T., Pérez, L.M.: Application of an obesity treatment protocol for 2 years. Nutricion hospitalaria 29(2), 300–304 (2014)
4. Catalán-Salgado, E.A., Zagal-Flores, R., Fernández, Y.T.T., Nieves, A.P.: Diet generator using genetic algorithms. Research in computing science 75, 71–77 (2014)
5. Ezquerra, E.A., Vázquez, J.M.C., Barrero, A.A.: Obesidad, síndrome metabólico y diabetes: implicaciones cardiovasculares y actuación terapéutica. Revista española de cardiología 61(7), 752–764 (2008)
6. Henríquez Sánchez, P., Doreste Alonso, J., Sevillano, L., Pilar, González, E., D., M., Valle, I., Mercedes, Martin López, G., Sosa Iglesias, I., Serra Majem, L.: Prevalencia de obesidad y sobrepeso en adolescentes canarios. relación con el desayuno y la actividad física. Medicina Clínica 130(16), 606–610 (2008)
7. Herrera, J.C.E., López, J.C.Á., Ramírez, L.G., Aguilar, R.J., Seguí, F.M., Quintanilla, R.H., Hernández, A.M.H., Lugo, E.C., Molina, H.A.L.: Comparación de métodos de estimación del gasto energético en reposo en adultos jóvenes de Yucatán, México. Revista Biomédica 30(3), 105–115 (2019)
8. INEGI, INSP: Encuesta nacional de salud y nutricó 2018 (ENSANUT, 2018). https://ensanut.insp.mx/encuestas/ensanut2018/doctos/ informes/ensanut_2018_presentacion_resultados.pdf (2018)
9. López-Jiménez, F., Cortés-Bergoderi, M.: Obesidad y corazón. Revista espanola de cardiologia 64(2), 140–149 (2011)
10. López López, M.K., Zamora Díaz, J.A.: Cálculo de una dieta balanceada mediante la aplicación de un algoritmo genético (2018)

11. Martínez, A.S., Martínez-Romillo, P.D., Tarrío, F.R.: Valoración del gasto energético en los niños. Implicaciones fisiológicas y clínicas. Métodos de medición. Anales de Pediatría 68(2), 165–180 (2008)
12. OMS: Obesidad y sobrepeso. https://www.who.int/es/news-room/fact-sheets/detail/obesity-and-overweight (2020)
13. OPS: La obesidad entre los niños y los adolescentes se ha multiplicado por 10 en los cuatro últimos decenios (2017)
14. Oviedo, G., Marcano, M., Morón de Salim, A., Solano, L.: Exceso de peso y patologías asociadas en mujeres adultas. Nutrición Hospitalaria 22(3), 358–362 (2007)
15. Arranz de la Peña, J., Parra Truyol, A.: Algoritmos genéticos. Universidad Carlos III (2007)
16. Rivero Gestal, M., Rabuñal, D.R., Dorado, J., Pazos, A.: Introducción a los algoritmos genéticos y la programación genética. Universidade da Coruña (2010)
17. Sánchez, O.C., Rocha-Díaz, J., Ramos-Aispuro, M.: Evaluación de los hábitos alimenticios y estado nutricional en adolescentes de sonora, méxico. Archivos en Medicina Familiar 10(1), 7–11 (2008)
18. Teijón, J.M.: Fundamentos de bioquímica metabólica, vol. 1 (2006)
19. Vargas Rivera, A.L.: Estudio del gasto calórico corporal mediante un sistema experto en nutrición resolviendo datos difusos. Fides et Ratio-Revista de Difusión cultural y científica de la Universidad La Salle en Bolivia 14(14), 127–144 (2017)
20. Wulandhari, L.A., Isa, S.M., et al.: Optimum nutrition intake from daily dietary recommendation for indonesian children using binary particle swarm optimization algorithm. Procedia Computer Science 157, 16–24 (2019)

ISSN 1870-4069

Análisis de las emisiones de NO2 durante la Jornada Nacional de Sana Distancia mediante imágenes Sentinel-5P

Julio Víctor Sánchez Hernández, Fernando Pech-May, Fernando Vera-Priego, David Salomón de la O Hidalgo, Luis Antonio López Goméz

Tecnológico Nacional de México Campus de los Ríos, Balancán, Tabasco México

hjulio288@gmail.com, fernando.pech@cinvestav.mx, fverapriego@gmail.com, d.delao@itsr.edu.mx, luisjezra@gmail.com

Resumen. La pandemia por COVID-19 ha afectado a millones de personas de todo el mundo. Distintos gobiernos han implementado medidas para reducir sus efectos. En México, se implementó la Jornada Nacional de Sana Distancia. Sin embargo, las medidas tomadas han ocasionado aspectos positivos y negativos en el medio ambiente y en la población tales como la reducción en las emisiones de dióxido de nitrógeno y los efectos económicos, respectivamente. En este artículo se analizan los niveles de dióxido de nitrógeno de los principales estados de la república mexicana utilizando imágenes satelitales multiespectrales obtenidas del satélite Sentinel-5P. Para evaluar la efectividad de la jornada nacional de sana distancia, se analizaron los niveles de emisión antes y durante las medidas tomadas por el gobierno.

Palabras clave: Sentinel-5P, JNS, GEE, NO2.

Analysis of NO2 Emissions during the National Day of Healthy Distance using Sentinel-5P images

Abstract. The COVID-19 pandemic has affected millions of people around the world. Different governments have implemented measures to reduce its effects. In Mexico, the National Day of Healthy Distance was implemented. However, the measures taken have caused positive and negative aspects for the environment and the population, such as the reduction in nitrogen dioxide emissions and the economic effects, respectively. This article analyzes the nitrogen dioxide levels of the main states of the Mexican Republic using multispectral satellite images obtained from the Sentinel-5P satellite. To evaluate the effectiveness of the national day of healthy distance, emission levels were analyzed before and during the measures taken by the government.

Keywords: Sentinel-5P, JNS, GEE, NO2.

1. Introducción

El nuevo virus SARS-COV-2 comúnmente denominado COVID-19, se ha propagado con gran rapidez afectando a millones de personas en casi todo el mundo. Para contener la transmisión de este mortal virus, se hanimplementado medidas estrictas que van desde el cierre de ciudades y países, hasta el paro de actividades no esenciales. El resultado de estas medidas trajo consigo un aspecto positivo para el medio ambiente, como descenso de las emisiones de dióxido de nitrógeno (NO2) en las principales ciudades del mundo.

El NO2 es un gas más pesado que el aire y se toma de referencia para medir los niveles de polución entre las diversas sustancias contaminantes que emiten los vehículos e industrias [9]. Algunos países como China e India son considerados como los principales emisores de contaminantes en el mundo; sin embargo, esto ha cambiado desde la propagación del COVID-19 [8,10]. En México, se implementó la Jornada Nacional de Sana Distancia (JNSD) con el objetivo de contrarrestar posibles contagios entre los mexicanos.En México, se implementó la Jornada Nacional de Sana Distancia (JNSD) con el objetivo de contrarrestar posibles contagios entre los mexicanos.

En este artículo se presenta una evaluación de las emisiones de NO2 en los principales estados de la República Mexicana mediante imágenes satelitales obtenidas del satélite Sentinel-5P, esta evaluación se realizó con fechas posteriores y dentro del marco de la JNSD.

2. Antecedentes

Dióxido de nitrógeno (NO2). Es un gas corrosivo, penetrante y oxidante formado por la combinación de nitrógeno y oxígeno [1]. Se produce naturalmente por incendios forestales, erupciones volcánicas o la descomposición de nitratos orgánicos; el volumen total que se produce de forma natural de este gas es menor a la que se produce por acciones humanas [6]. La mayor parte de su origen es por la oxidación del monóxido de nitrógeno (NO), que se produce en la combustión de los motores de los vehículos; al entrar en contacto con la atmosfera se oxida y se convierte en NO2 [6]. La exposición contínua al NO2 se relaciona con diversas enfermedades respiratorias como disminución de la capacidad pulmonar, asma, entre otras. Las personas con enfermedades respiratorias crónicas son vulnerables y susceptibles a sufrir severos problemas a causa de este gas. Algunos estudios apuntan que el incremento de NO2 en las ciudades es uno de los factores de mortalidad [1,6].

Google Earth Engine (GEE). Es una plataforma online gratuita que trabaja con el lenguaje de programación JavaScript; pone a disposición una gran cantidad de imágenes satelitales, actuales e históricas e información completa sobre las misiones espaciales Landsat, Copernicus Sentinel, entre otros [5]. Posee capacidad de análisis a escala planetaria mediante algoritmos de aprendizaje automático para el monitoreo ambiental. Fue creado con la intención de monitorear y medir los cambios que sufre el medio ambiente, así como dirigir recursos

para actuar ante desastres naturales futuros y evitar degradaciones forestales. Ofrece la estructura computacional de Google para analizar todas las imágenes y, por otro lado, puede utilizarse para la medición, reporte y verificación de las iniciativas que buscan detener la deforestación en el planeta [2].

Sentinel-5P. Es el primer satélite del programa Copernicus encargada de monitorear la atmosfera terrestre a escala global; lleva a bordo un sensor denominado TROPOspheric Monitoring Instrument (TROPOMI) que mapea una multitud de gases traza como dióxido de nitrógeno y azufre, ozono, entre otros; los cuales afectan el aire que respiramos y, como consecuencia, nuestra salud y clima [3]. Es considerado una herramienta muy importante en el campo de la investigación, ya que contribuye a la obtención de datos atmosféricos homogéneos y precisos para su aplicación en diversas áreas. Tal es el caso de Muhammad *et al.*, [7] que presentan un estudio donde recopilaron datos ambientales del Sentinel-5P publicados por la NASA y ESA antes y después de las medidas de restricción a causa del COVID-19 en algunos epicentros como Wuhan, Italia, España y EE. UU.

3. Materiales y métodos

La metodología aplicada para esta investigación se dividió en cuatro etapas (ver Figura 1): 1) determinar las áreas de estudio; 2) determinar las series temporales; 3) adquirir las colecciones de imágenes Sentinel-5P; 4) procesar, calcular y representar la información obtenida del satélite.

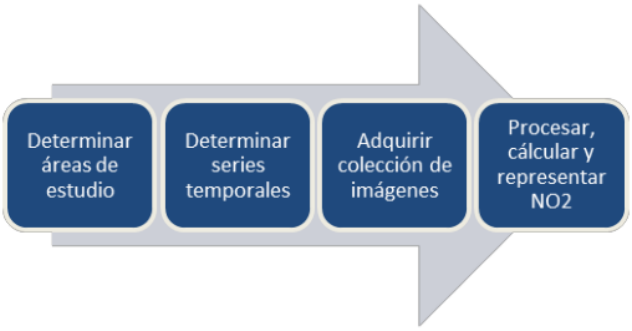


Fig. 1. Metodología propuesta para el análisis de emisiones de NO2.

Áreas de estudio. Las áreas de estudios para esta investigación se escogieron con base a características que poseen en común cada uno de los estados; una de estas características es la presencia de industrias. Los estados elegidos como zonas de estudio fueron (ver Figura 2): 1) Ciudad de México, capital de la Republica Méxicana y centro político y económico del país. Su área metropolitana es la novena más poblada del mundo, y la más poblada de Norteamérica; 2) Jalisco,

es la cuarta entidad federativa más poblada de México y uno de los Estados más desarrollados en el país en cuanto a actividades económicas, comerciales y culturales; 3) Nuevo León, es uno de los estados con mejores estándares de vida, principalmente por las condiciones de seguridad pública, eventos culturales, hospitales e instituciones de salud, opciones de entretenimiento, ingreso per cápita, instituciones educativas y ubicación geográfica; 4) Puebla, uno de los estados más poblados del país por detrás del Estado de México, Veracruz, Jalisco y Ciudad de México, el mayor sector de la economía poblana es el de la industria manufacturera, que contempla la maquila, especialmente de productos textiles; 5) Tabasco, con gran riqueza en recursos naturales, además, ofrece una excelente oportunidad para emprender negocios en la entidad. La disponibilidad de su mano de obra y su cercanía con los principales puertos del país y centros de consumo, brindan una inmejorable oportunidad para invertir.

Fig. 2. Metodología propuesta para el análisis de emisiones de NO2.

Series temporales. Para llevar a cabo la evaluación de las emisiones de NO2 en las áreas de estudio, se establecieron dos series temporales. La primera temporada de estudio corresponde del 01 de enero al 22 de marzo de 2020, es decir, los dias previos a la implementación de la JNSD; la segunda serie temporal corresponde a los dias que se implementó la JNSD, del 23 de marzo al 30 de mayo de 2020.

Adquisición de imágenes. Los satélites Sentinel de la ESA recopilan gran variedad de imágenes y datos atmosféricos a escala global. Muchos de estos datos, que cubren un amplio período de tiempo, están disponibles para el público a través de diversas plataformas gratuitas como GEE. Se descargaron datos de las emisiones de NO2 a través de la plataforma GEE. Los datos recopilados fueron filtrados con base a las series de tiempo establecidas y las zonas de estudios previamente seleccionadas. Para descargar los datos de NO2 se utilizó la banda *NO2_column_number_density* que se encuentra en el sensor TROPOMI del

Sentinel-5P; esta banda es la encargada de almacenar los datos relacionados con las emisiones totales de NO2.

Procesamiento y cálculo del NO2. Posterior a la selección y filtrado de la colección de imágenes, se calcularon los valores de las emisiones de NO2. Las columnas de NO2 se derivan del espectrómetro UVIS[1] de TROPOMI, medidas de radiación solar retro dispersadas en el rango de longitud de onda de 405–465 nm; usa un sistema de procesamiento que se basa en DOMINO[2] y QA4ECV. La base para el procesamiento es un sistema de modelado de recuperación - asimilación que utiliza un modelo tridimensional global de transporte químico TM5; la recuperación consiste en un procesamiento de 3 pasos [4]:

- Recuperación de la densidad de columna inclinada (SCD)[3] a partir de los espectros de irradiancia.
- Separación de la densidad de columna inclinada en la parte estratosférica y troposférica sobre la base de información proveniente de un sistema de asimilación de datos.
- Conversión de la densidad de columna inclina estratosférica y troposférica en densidad.

4. Resultados

La implementación de la JNSD redujo las actividades de transporte, industrias, entre otras. Trajo consigo la reducción en el consumo de energía y una menor demanda de petróleo; estos cambios tuvieron un impacto positivo y significativo en la calidad del medio ambiente. A continuación, se presentan una serie de mapas y gráficas de las emisiones de NO2 antes y durante la JNSD en los estados en estudio. Dichos datos fueron recolectados del satélite Sentinel-5P y procesados en la plataforma GEE.

4.1. Emisiones de NO2 previo a la JNSD

En la secuencia de imágenes satelitales que se muestran a continuación se ilustran las emisiones de NO2 previo a la implementación de la JNSD:

- La Figura 3 hace referencia a la CDMX. Se encontró una concentración media mínima de NO2 de 0.0000994 Mol/M2 y máxima de 0.000359 Mol/M2; el punto más denso de concentración fue en la zona norte del estado.
- La Figura 5 corresponde al estado de Nuevo León; se obtuvo una concentración media mínima de NO2 de 0.0000460 Mol/M2 y máxima de 0.000139 Mol/M2; el punto más denso de concentración fue en la capital del estado (Monterrey).

[1] UV Visible
[2] Algoritmo de recuperación de las columnas de NO2
[3] Proporciona la concentración a un determinado ángulo justo por encima del punto de observación, no proporciona la concentración total

- La Figura 4corresponde al estado de Jalisco; presentó una concentración mínima media de 0.0000366 Mol/M2 y máxima de 0.00008 Mol/M2; el punto más denso de emisiones fue en la ciudad de Guadalajara.
- La Figura 6 se refiere al estado de Tabasco; los valores medios mínimos (0.0000387 Mol/M2) y máximos (0.0000476 Mol/M2) de las emisiones de NO2 en el estado.
- En la Figura 7 se muestran los valores medios de las emisiones de NO2 en el estado de Puebla; tuvo un valor mínimo de 0.0000394 Mol/M2 y máximo de 0.0000789 Mol/M2.

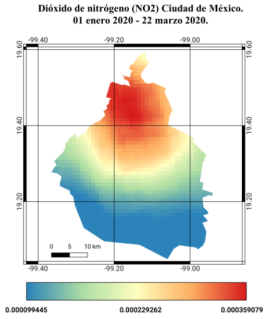


Fig. 3. Imagen satelital de las emisiones de NO2 en CDMX.

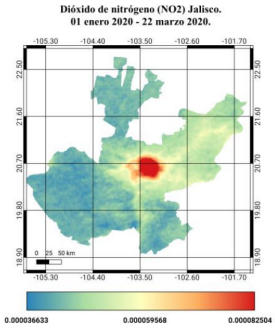


Fig. 4. Imagen satelital de las emisiones de NO2 en Jalisco.

4.2. Emisiones de NO2 durante la JNSD

A continuación, se presentan una serie de imágenes satelitales que ilustran las emisiones de NO2 durante la implementación de la JNSD. En la Figura 8 se presentan los valores medios mínimos y máximos de las emisiones de NO2 en la Ciudad de México, el valor mínimo corresponde a 0.0000788 Mol/M2 y el máximo fue de 0.000178 Mol/M2. En la Figura 9 se muestran las emisiones en el

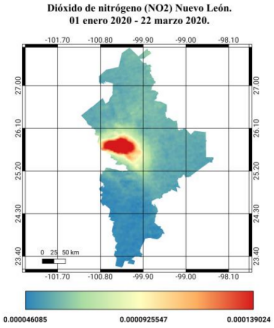


Fig. 5. Imagen satelital de las emisiones de NO2 en Nuevo León.

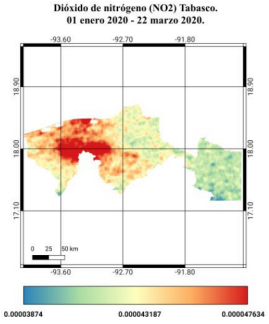


Fig. 6. Imagen satelital de las emisiones de NO2 en Tabasco.

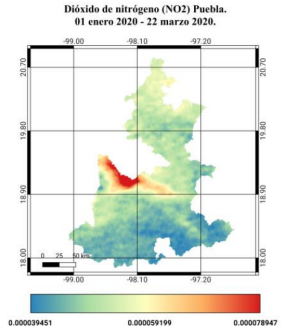


Fig. 7. Imagen satelital de las emisiones de NO2 en Puebla.

estado de Nuevo León, el valor medio mínimo de 0.0000481 Mol/M2 y un valor medio máximo de 0.0000882 Mol/M2. En la Figura 10 se muestran las emisiones registradas en el estado de Jalisco, con un valor medio mínimo de 0.0000366 Mol/M2 y un valor máximo de 0.0000825 Mol/M2. En la Figura 11 se muestran los valores medios mínimos y máximos de las emisiones de NO2 en el estado de Tabasco, los valores registrados fueron 0.0000521 Mol/M2 como mínima y

0.0000641 Mol/M2 como máxima. Finalmente, en la Figura 12 se muestran los registros obtenidos en el estado de Puebla, el valor medio mínimo obtenido fue de 0.0000493 Mol/M2 y el máximo registrado fue de 0.0000808 Mol/M2.

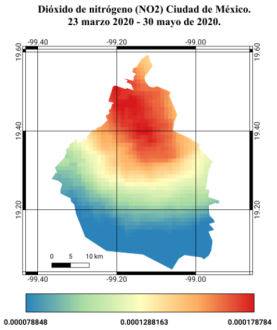


Fig. 8. Emisiones de NO2 en CDMX durante la JNSD.

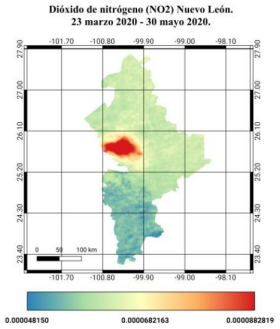


Fig. 9. Emisiones de NO2 en Nuevo León durante la JNSD.

4.3. Variación de las emisiones de NO2

A continuación, se presentan las gráficas comparativas de los valores medios mínimos y máximos previo y durante la implementación de la JNSD.

En la Figura 13 se presenta la gráfica de los valores medios mínimos por estado, se puede observar que la CDMX presentó un descenso significativo en las emisiones mínimas de NO2, por otra parte, el estado de Jalisco mantuvo constante la emisión de NO2 y el resto de los estados en estudio presentaron un incremento en los valores medios mínimos de emisiones de NO2.

En la Figura 14 se muestra la gráfica comparativa de los valores medios máximos de NO2 por estado, en ella se aprecia que la CDMX y Nuevo León tuvieron una baja en sus niveles máximos de emisión de NO2; Jalisco y Puebla mantuvieron sus niveles máximos de emisión, por otro lado, Tabasco incrementó sus niveles máximos de emisión de NO2.

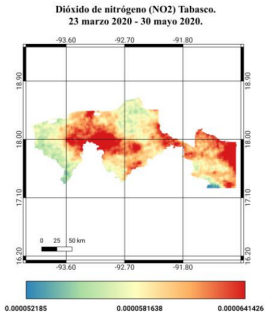


Fig. 10. Emisiones de NO2 en Tabasco durante la JNSD.

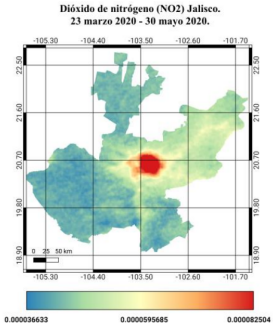


Fig. 11. Emisiones de NO2 en Jalisco durante la JNSD.

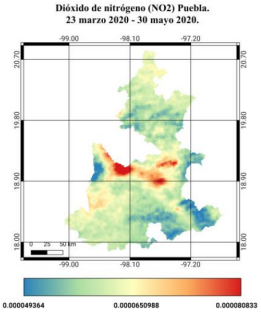


Fig. 12. Emisiones de NO2 en Puebla durante la JNSD.

5. Conclusiones

SARS-CoV-2 es una grave pandemia que amenaza y afecta la salud pública y las actividades económicas de diferentes países del mundo. Sin embargo, trajo consigo un aspecto positivo para nuestro planeta, la reducción de la contaminación. Este impacto positivo en el medio ambiente puede ser temporal, pero los

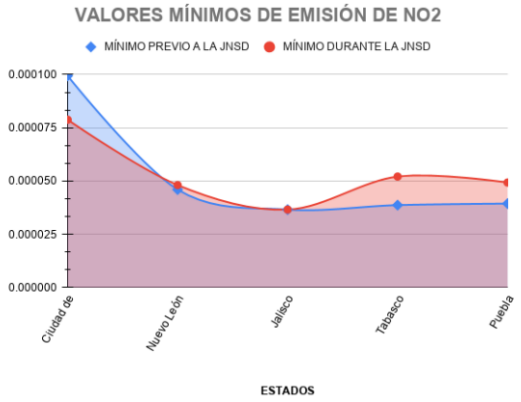


Fig. 13. Comparación de emisiones mínimas de NO2 por estado.

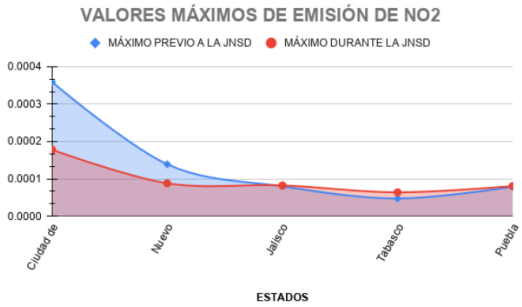


Fig. 14. Comparación de emisiones máximas de NO2 por estado.

gobiernos y las personas deberían aprender de estas medidas de mitigación para saber cómo reducir la contaminación a largo plazo.

El proyecto Sentinel-5P de la ESA, produce datos abiertos de alta resolución espacial, disponibles en línea y casi en tiempo real con el objetivo de monitorear la calidad del aire a escala local, nacional o global, un argumento aún más importante de la fiabilidad de estos datos es la cobertura global diaria, que proporciona una superficie continua de datos, mostrando la distribución espacial de diferentes contaminantes durante un tiempo específico. Sin embargo, existen algunas limitaciones en el uso de datos de Sentinel-5P, como la presencia de nubes y el número de observaciones por día, que no son suficientes para un monitoreo detallado, ya que los contaminantes pueden dispersarse fácilmente en la atmósfera.

Por otro lado, los resultados expuestos anteriormente demuestran el descenso de los niveles de NO2 en los estados de CDMX, Nuevo León y Jalisco, lo cual quiere decir que su población ha seguido las indicaciones del gobierno federal. Sin embargo, según los datos recabados los estados de Puebla y Tabasco

no mostraron cambios significativos en sus niveles de NO2, lo que nos hace concluir que en estos estados no se siguieron las normas de sana distancia y como consecuencias los números de contagios han ido en aumento.

Referencias

1. Centro de Monitoreo de Calidad del Aire del Estado de Querétaro: Dióxido de nitrógeno (2017), http://www.cemcaq.mx/contaminacion/bioxido-de-nitrogeno-no2
2. Ruiz de Eguino, I.A.: Towards ndvi-based continuous monitoring for energy transport infrastructure maintenance (2018)
3. European Space Agency: Sentinel-5P (2017), https://sentinels.copernicus.eu/web/sentinel/missions/sentinel-5p
4. Forero Castro, D.D.: Metodología para la incorporación de datos del sensor tropomi del satélite sentinel 5-p al monitoreo de la calidad del aire en Bogotá D.C., https://repository.udistrital.edu.co/bitstream/handle/11349/23649/ForeroCastroDiegoDaniel2019.pdf?sequence=1&isAllowed=y
5. Gorelick, N., Hancher, M., Dixon, M., Ilyushchenko, S., Thau, D., Moore, R.: Google earth engine: Planetary-scale geospatial analysis for everyone. Remote Sensing of Environment 202, 18–27 (2017), https://www.sciencedirect.com/science/article/pii/S0034425717302900
6. Instituto para la Salud Geoambiental: Dióxido de nitrógeno no2 (2020), https://www.saludgeoambiental.org/dioxido-nitrogeno-no2
7. Muhammad, S., Long, X., Salman, M.: COVID-19 pandemic and environmental pollution: A blessing in disguise? Science of The Total Environment 728, 138820 (2020), https://www.sciencedirect.com/science/article/pii/S0048969720323378
8. National Center for Biotechnology Information: Wuhan seafood market pneumonia virus isolate wuhan-hu-1, complete genome (2019), https://www.ncbi.nlm.nih.gov/nuccore/MN908947.2
9. Valenzuela, A.: ¿Qué es el dióxido de nitrógeno? (2015), https://www.rtve.es/noticias/20150118/dioxido-nitrogeno/1083180.shtml
10. World Health Organization: Novel coronavirus (2019-ncov): situation report (2020), https://apps.who.int/iris/bitstream/handle/10665/330775/nCoVsitrep30Jan2020-eng.pdf?sequence=1&isAllowed=y

ISSN 1870-4069

Comparative Study of Optimizers in the Training of a Convolutional Neural Network in a Binary Recognition Model

Marco López-Sánchez, José Hernández-Torruco, Betania Hernández-Ocaña, Oscar Chávez-Bosquez

Universidad Juárez Autónoma de Tabasco,
División Académica de Ciencias y Tecnologías de la Información,
Cunduacán, Tabasco,
Mexico

{marco.lopezsanchez,oscar.chavez,jose.hernandezt,
betania.hernandez}@ujat.mx

Abstract. Comparative study of optimizers in the training of a convolutional neural network in a binary recognition model. In Machine and Deep learning, the optimizer selection is a crucial step, as the model performance depends heavily on the optimizer selected. Deep neural network architectures often have multiple layers of representations used to learn from training data automatically. Hyperparameter calibration plays an essential role in the learning process. As the parameter values may vary depending on the data set, we want to obtain an essential improvement in the model performance in the training phase. This work's main objective was to determine which optimizer can obtain the best accuracy in the training phase of a convolutional neural network used to perform binary classification using the public dataset Dogs vs. Cats, consisting of a total of 25,000 images. The compared optimizers are SGD, Adam, and RMSprop. The model architecture developed consists of three convolution layers with their corresponding maxpooling layer. As a result, experiments show that the Adam optimizer offers better performance. These results suggest that the use of Adam to train binary convolutional neural networks should be considered.

Keywords: Image classification, optimizer, deep learning, convolutional neural networks.

1 Introduction

The implementation of computer vision has attracted the interest of researchers within the area of computer science, as Deep learning has become a promising area in the field of computer vision [10]. Deep learning uses Artificial neural networks internally as the main algorithm to perform predictions, classification,

and so on. Image classification, covering a diverse range of applications including biometrics [12], video surveillance [18] and medical research [14].

Among the most promising algorithms, the convolutional neural network (CNN) technique [15], is the most widely used deep learning algorithm. In this paper, we describe a binary classification problem and analyze the Adam, RMSprop, and SGD optimizers, which are used to train deep learning neural networks. ADAM is the most recent and represents an improvement of other optimizers, but it is not always the one that generates the best model. Following, we present key concepts and optimization techniques, followed by a brief description of neural convolution networks. The rest of the work is organized in the following order: Section 2 materials and methods in section 2. The results of the experiments are presented in section 3. Finally, section 4 concludes the article.

1.1 Artificial Neural Networks

Artificial neural networks (ANN) aim to perform computational tasks using a large number of simple interconnected processing units called neurons or nodes. The connections between neurons are associated with parameters called weights that can be modified, through a training process, to associate the desired output to a specific input [4]. It can be described as a directed graph in which each node performs a transfer function of the form [23].

Hyperparameters are a particular set of parameters used in the learning process during training. Hyperparameters are vital since they directly impact the neural network's training phase and hence in the model performance. There are several hyperparameter optimization techniques, but it is the data scientist's responsibility to initialize these values manually. The parameters might be an integer or a continuous or categorical variable which ranges from the lower to upper bound values [22].

The basic hyperparameters of an ANN are:

- Number of hidden layers: Neural networks with single-layer tunable parameters are very limited in what they can do, as is the case with perceptrons. Therefore, it is natural to expand a neural network's capacity by adding additional layers of neurons. From outside the network, only the first and last layers of a multi-layer network are visible: the input and output layers. All other layers are "hidden" layers". The additional layers are therefore called hidden layers because they are not visible from the outside [2].
- Number of hidden units: Each hidden layers can have a different number of neurons.
- Learning rate: The learning rate is a small number, often between 0.00001 and 0.05, which affects the magnitude that is added to the current weight of an edge in order to train the model with these updated weights [5]. In nearly all gradient descent algorithms, the choice of learning rate remains central to efficiency. Yoshua Bengio [1] claims that it is often the single most crucial hyper-parameter and that it always should be tuned.

- Number of epochs: One epoch means that each sample in the training dataset has had an opportunity to update the internal model parameters. An epoch is comprised of one or more batches. For example, an epoch that has one batch is called the batch gradient descent learning algorithm [3].
- Batch size: The batch size is a hyperparameter that defines the number of samples to work through before updating the internal model parameters [3].
- Loss function: This is a tool to measure a decision algorithm's performance (or classifier). The loss function measures each classifier action's cost and converts an error probability error into a decision. This approach allows us to handle situations where particular classification mistakes have to be considered differently from the others [4].
- Activation function: This is critical as images and features within an image are highly non-linear problems, and most other functions within Convolutional neural networks (Conv2D, pooling, fully connected layers, and so on) generate only linear transformations. The activation function generates the non-linearity while mapping input values to its ranges. Many activation functions are used, but the most common ones are Sigmoid, Tanh, ReLU.

An optimizer's role is to update the weight parameters then to minimize the error function or loss function, where the error is the difference between the actual value and the predicted value. Before starting the training phase, the optimizer that carries out this task must be selected along with its specific algorithmic hyperparameters. The optimizers compared in this study are:

SGD The Stochastic Gradient Descent algorithm [19] is one of the earliest methods used for training neural networks. The main advantage of these methods is the simplicity of each iteration, both in generating the search direction and in performing the update of variables [17].

RMSprop It was introduced to address the monotonically decreasing learning rate problem. This problem is present in the AdaGrad optimizer [11]; it uses exponential decay in the first step.

Adam The Adam optimization [13] algorithm was introduced to combine the benefits of AdaGrad, and RMSProp algorithms.

The performance metrics used in this work are:

- Accuracy: It is the most used metric to summarize the performance of a supervised learning model, as it indicates the proportion of examples that a classifier can classify correctly, usually indicated as a percentage [2].
- Confusion matrix: In its simplest form, a confusion matrix (also called an error matrix) is a type of contingency table with two rows and two columns that contains the numbers of false positives, false negatives, true positives, and true negatives [5].
- ROC curve: The ROC (receiver operating characteristic) curve is a curve that plots the TPR (true positive rate, i.e., the recall) against the FPR (false positive rate) [5].

1.2 Convolutional Neural Networks

A convolutional neural network (CNN) is a deep learning algorithm specialized in image classification. The core element of CNNs is the processing of data through the convolution operation. Convolution of any signal that is combined with another one produces a third signal that may reveal more information about the signal than the original signal itself [6].

In 1990, LeCun et al. [16] published the seminal paper fundamenting the modern framework of a CNN, and later improved it in [15]. The specific hyperparameters of CNN are [22]:

- Number of convolutional layers. This layer is the primary building block of a convolutional neural network, which determines the output of associated.
- Number of pooling layers. The primary function of the Pool layer is to progressively reduce the spatial size (i.e., width and height) of the input volume. Typically, we use a pool size of 2×2, although deeper CNNs that use larger input images (> 200 pixels) may use a 3×3 pool size early in the network architecture.
- Step size handled by the stride. It is the number of pixels to move to define the local receptive field for a filter–a stride of 1 means to move across and down a single pixel. The value of stride should not be too small or too large, and it is almost always symmetric in the dimensions of height and width.

A convolutional neural network, also known as ConvNet or CNN, is a variant of the artificial neural network, which specializes in emulating functionality and behavior of our visual cortex [21]. CNN has produced excellent results in fields like image classification [15], speech recognition [9] and object identification [7].

Typical CNN architectures stack a few convolutional layers (each one generally followed by a ReLU layer), then a pooling layer, then another few convolutional layers (+ReLU), then another pooling layer, and so on [8].

When CNNs are applied to images, consecutive convolutional layers learn progressively abstract features. Figure 1 depicts this process, where the first layers take care of extracting characteristics such as the edges or corners of an image. Once these edges are detected, they are used to detect shapes in the subsequent layers. When the shapes are detected, high-level characteristics are detected, such as a wheel in an image where a car appears. The last layers are fully connected and take care of giving a prediction from these last extracted characteristics.

2 Materials and methods

2.1 Dataset

The Dogs vs. Cats dataset was originally collected by Microsoft and consists of more than 3 million images, of which 25,000 have been publicly released[1]. It is

[1] https://www.microsoft.com/en-us/download/confirmation.aspx?id=54765

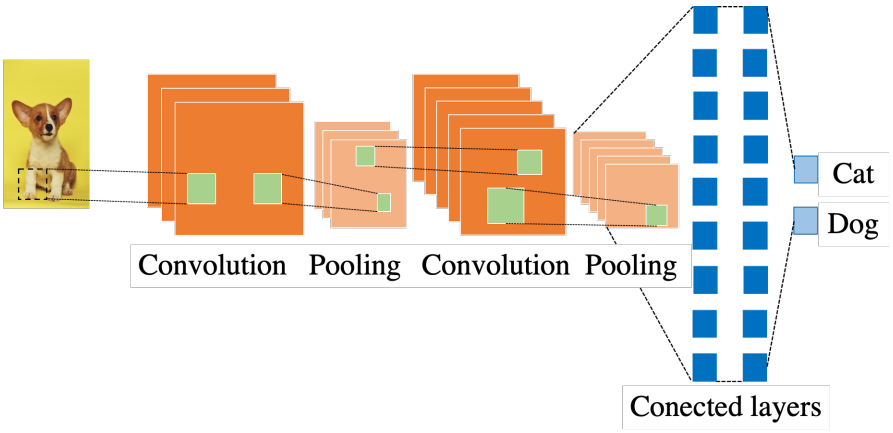


Fig. 1. Example of a convolutional neural network.

worth mentioning that there is a smaller dataset (3,000 images) derived from this one, that is more commonly used in the literature.

Data are the images, which are stored in 2 directories, one for the pictures of cats and the other one for pictures of dogs. The dataset is balanced, i.e., it has the same number of images of cats and dogs. The goal is to build a machine learning algorithm capable of correctly detecting the animal (dog or cat) in the images.

2.2 Experimental setup

The CNN model proposed in this paper is used to predict the correct class of each item in the dataset. The data were split into 70 % training, 15 % for validation, and 15 % for testing. The total number of convolutional layers is 3*stacks. Each stack has a convolutional layer, a batch normalization layer, and a ReLU layer. The image size is 150∗150∗3, where 150∗150 is the size in pixels, and the number 3 represents the depth of the image.

The hyperparameter configuration was:

- Learning rate: 0.00001,
- Number of epochs: 30,
- Steps per epoch: 70,
- Loss function: Binary crossentropy,
- Activation functions: ReLU and Sigmoid.

The life cycle of a model provides the backbone for modeling a dataset. The life cycle used in this work includes [20]:

1. Gather the dataset: download a zip file containing the dataset, unzip it and discard the corrupt files.

2. Split the dataset: divide the data set into three parts:
   – Training set: the set of samples used to train the model.
   – Validating set: the set of samples that we are going to use to adjust the parameters of the classifier.
   – Testing set: the set of samples that we will use only to evaluate the performance of the classifier.

To the training set we assign 70 % of the images, to the validating 15 % and to the testing 15 %. Train network: In this step, we define the architecture of our network configuring each of its layers. We also configure the hyperparameters, then compile the model and feed it with the training set. Evaluate: To verify the result obtained in training, we use the confusion matrix and the ROC curve. Make predictions: We provide a simple user interface so that the user can load an image and test the prediction in a more intuitive way.

3 Results

We conducted experiments with the dataset to determine the behavior of each optimizer. To test the performance of each optimizer in all the experiments, the default settings of each hyperparameter were chosen. We have included an early stopping in the training phase to stop the training loop after 5 iterations with no improvement. Table 1 shows the main performance metrics on each optimizer. Table 1 shows that the highest accuracy was achieved by the Adam optimizer, while RMSprop was second with a small gap. SGD fall behind with a weak accuracy.

Table 1. Model results.

Optimizer	Loss	Accuracy
SGD	0.6926	0.5053
RMSprop	0.4520	0.7868
Adam	0.4227	0.8017

Figure 2 shows the performance of each model in the training phase on the training and validation subsets. We highlight that SGD stops training at epoch 12, according to the early stopping configuration. On the other side, RMSprop and Adam continue the training until the 30 configured epochs. SGD validation curve shows an erratic behavior, while RMSprop and Adam curves shows the slope of the learning process.

The confusion matrix of each optimizer is presented in Figure 3. We can notice that SGD has the lower performance, misclassifying most of the images. On the other hand, RMSprop and Adam performed quite well in the classification of cats and dogs.

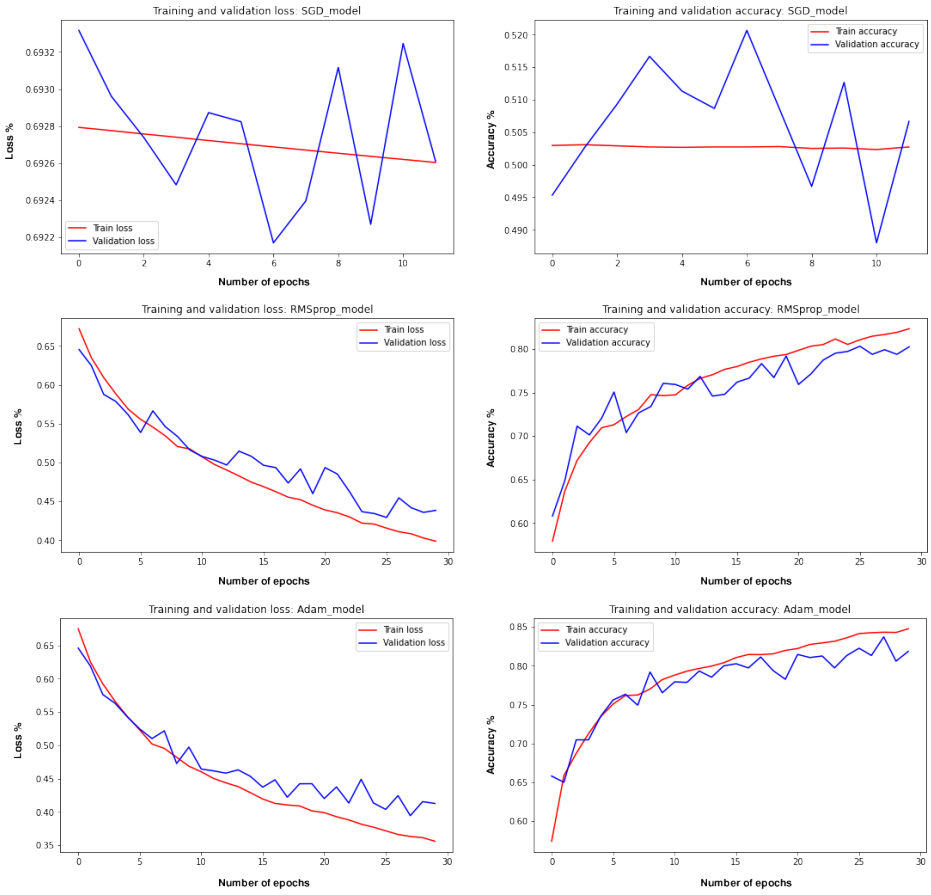


Fig. 2. Optimizers performance.

Figure 4 shows the graphical representation with normalized values of the confusion matrix of each optimizer. In these plots the difference between the correct classifications (top left and bottom right of each plot) against misclassifications (bottom left and top right) is clear. For example, we can see that SGD virtually classifies all the images as dogs, having a high rate with dog images but a low rate classifying cat images.

In Figure 5 we can see the area under the curve (AUC) of the ROC curve of each optimizer, computing the false-positive rate and the true-positive rate of each model. The AUC measures how much the model is capable of distinguishing between cats and dogs. We can notice the flattened curve in the SGD optimizer, indicating a poor classification performance. The ROC curve for RMSprop and Adam is similar, along with their AUC. Both optimizers performed well in the classification task.

		SGD optimizer		RMSprop optimizer		Adam optimizer	
		Predicted value		Predicted value		Predicted value	
		Cat	Dog	Cat	Dog	Cat	Dog
Real value	Cat	38	1838	1392	484	1624	252
	Dog	18	1858	316	1560	492	1384

Fig. 3. Confusion matrix of each optimizer.

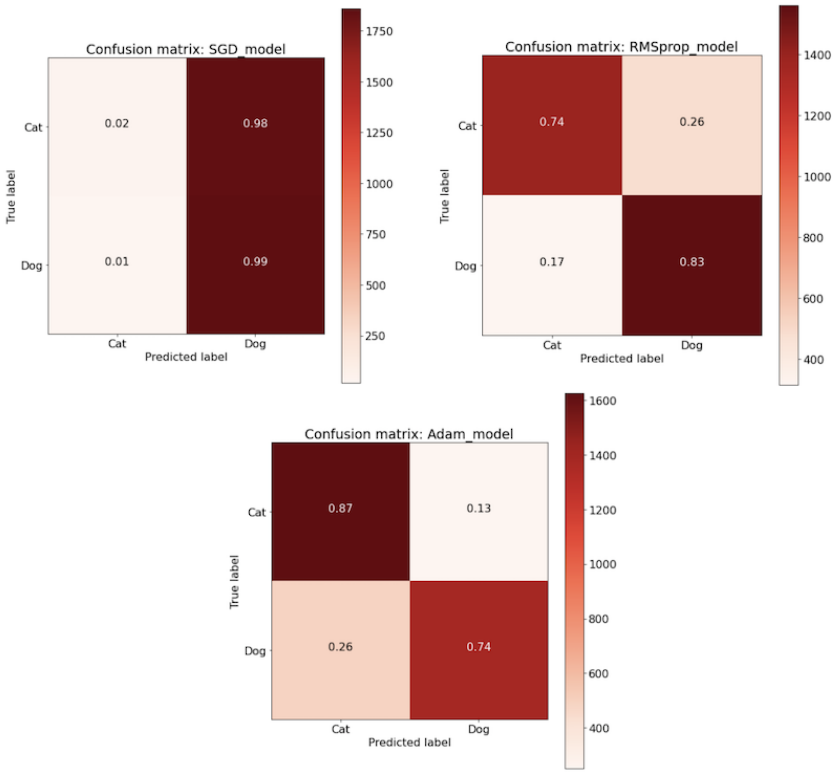


Fig. 4. Graphical representation of the confusion matrix.

4 Conclusions

Selecting an optimizer for training a neural network is a challenging task. The role of an optimizer is to update the weight parameters so the network can minimize the error or loss function, where the error is the difference of actual value and the predicted value.

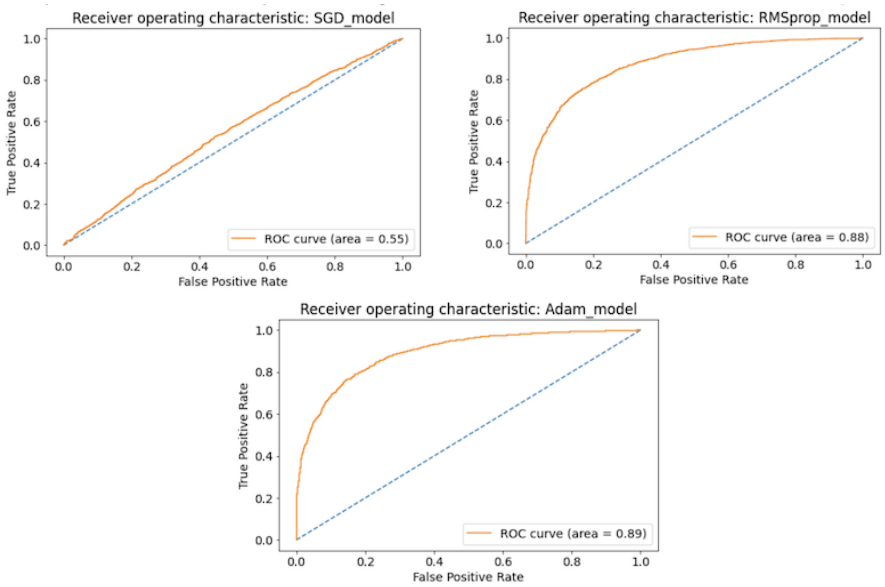


Fig. 5. ROC curve of each optimizer.

In this work, we test three optimizers: SGD, RMSprop, and Adam, creating a model for binary classification in the Dogs vs. Cats dataset. In our experiments, the Adam optimizer obtained the best performance, according to the model's accuracy and the AUC of the ROC curve. This may be due to the fact that Adam is an adaptative learning rate method, as it computes individual learning rates for diferent parameters. Adam has also the best performance in terms of speed of training.

Each optimizer has several parameters that have a direct impact on the performance of the generated model. In a future work, we consider the tuning of these parameters and using more and diverse optimizers.

Acknowledgments. To the Consejo Nacional de Ciencia y Tecnología (CONACYT) for supporting the Doctoral program in Computer Science at the Universidad Juárez Autónoma de Tabasco.

References

1. Bengio, Y.: Practical recommendations for gradient-based training of deep architectures. In: Neural networks: Tricks of the trade, pp. 437–478. Springer (2012)
2. Berzal, F.: Redes neuronales & Deep Learning: Volumen II. Edición Independiente, 1era. edn. (2019)
3. Brownlee, J.: Master Machine Learning Algorithms: Discover How They Work and Implement Them From Scratch. Jason Brownlee, 1st edn. (2016), https://books.google.com.mx/books?id=PCJnAQAACAAJ

4. Camastra, F., Vinciarelli, A.: Machine learning for audio, image and video analysis: theory and applications. Springer (2015)
5. Campesato, O.: Artificial Intelligence, Machine Learning, and Deep Learning. Stylus Publishing, LLC (2020)
6. El-Amir, H., Hamdy, M.: Deep Learning Pipeline: Building a Deep Learning Model with TensorFlow. Apress (2019)
7. Erhan, D., Szegedy, C., Toshev, A., Anguelov, D.: Scalable object detection using deep neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2147–2154 (2014)
8. Géron, A.: Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. O'Reilly Media (2019)
9. Guiming, D., Xia, W., Guangyan, W., Yan, Z., Dan, L.: Speech recognition based on convolutional neural networks. In: 2016 IEEE International Conference on Signal and Image Processing (ICSIP). pp. 708–711. IEEE (2016)
10. Hinton, G., Osindero, S., Teh, Y.W.: A fast learning algorithm for deep belief nets. Neural Computation 18(7), 1527–1554 (2006)
11. Hinton, G., Srivastava, N., Swersky, K.: Neural networks for machine learning lecture: Lecture 6a overview of mini-batch gradient descent. Coursera (2012), online
12. Jaseena, K., Kovoor, B.: A survey on deep learning techniques for big data in biometrics. International Journal of Advanced Research in Computer Science 9(1), 12–17 (2018)
13. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
14. Lakhani, P., Sundaram, B.: Deep learning at chest radiography: automated classification of pulmonary tuberculosis by using convolutional neural networks. Radiology 284(2), 574–582 (2017)
15. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature 521, 436–444 (2015)
16. LeCun, Y., Boser, B.E., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W.E., Jackel, L.D.: Handwritten digit recognition with a back-propagation network. In: Advances in neural information processing systems. pp. 396–404 (1990)
17. Nesterov, Y.: Efficiency of coordinate descent methods on huge-scale optimization problems. SIAM Journal on Optimization 22(2), 341–362 (2012)
18. Ojha, S., Sakhare, S.: Image processing techniques for object tracking in video surveillance-a survey. In: 2015 International Conference on Pervasive Computing (ICPC). pp. 1–6. IEEE (2015)
19. Robbins, H., Monro, S.: A stochastic approximation method. The annals of mathematical statistics pp. 400–407 (1951)
20. Rosebrock, A.: Deep Learning for Computer Vision with Python: Starter Bundle. PyImageSearch (2017)
21. Sarkar, D., Bali, R., Sharma, T.: Practical machine learning with python. A problem-solvers guide to building real-world intelligent systems. Apress, Berkely (2018)
22. Victoria, A.H., Maragatham, G.: Automatic tuning of hyperparameters using bayesian optimization. Evolving Systems (2020)
23. Yao, X.: Evolving artificial neural networks. Proceedings of the IEEE 87(9), 1423–1447 (1999)

ISSN 1870-4069

Mapeo de inundaciones utilizando imágenes satelitales SAR en Google Earth Engine

Julio Víctor Sánchez Hernández, Fernando Pech-May,
Honorio Guadalupe Sánchez Jacinto, Jorge Magaña-Govea

Tecnológico Nacional de México Campus de los Ríos,
Balancán, Tabasco,
México

hjulio288@gmail.com, fernando.pech@cinvestav.mx,
honoriosanchezjacin@gmail.com, totosaus@hotmail.com

Resumen. Tabasco es una entidad que padece de inundaciones de manera recurrente provocando pérdidas y efectos negativos en el sector rural, urbano, ganadero, agrícola y de servicios. Por otra parte, existen programas satelitales que proporcionan gran cantidad de datos precisos de la superficie terrestre, así como herramientas para el procesamiento de información geoespacial que son de gran utilidad para el monitoreo ambiental y forestal, efectos del cambio climático, análisis de riesgos, desastres naturales, entre otras. En este artículo se presenta el análisis de las áreas inundadas en la región Ríos del estado de Tabasco mediante imágenes satelitales Sentinel-1A y Sentinel-2A para el mapeo e identificación de áreas afectadas provocadas por las inundaciones presentadas en el mes de noviembre de 2020. Los resultados obtenidos muestran que más de 30,000 hectáreas de extensión territorial del área de estudio fueron afectadas con las inundaciones causando perdidas en el sector agrícola, ganadero y dejando incomunicadas localidades.

Palabras clave: Percepción remota, Sentinel-1A, radar de apertura sintética.

Flood Mapping using SAR Satellite Images in Google Earth Engine

Abstract. Tabasco is an entity that suffers recurrent floods causing losses and negative effects in rural, urban, livestock, agricultural and service sectors. There are satellite programs that provide a large amount of accurate data on the earth's surface and tools for processing geospatial information that are very useful for environmental and forest monitoring, effects of climate change, risk analysis, natural disasters, etc. This paper presents the analysis of flooded areas in the region of the state of Tabasco through Sentinel-1A and Sentinel2A satellite images to mapping and identification of affected areas caused by the floods in November 2020.

The results show that more than 30,000 hectares were affected by the floods, causing losses in the agricultural and livestock sectors and leaving localities isolated.

Keywords: Remote sensing, Sentinel-1A, synthetic aperture radar.

1. Introducción

Las inundaciones son eventos que se producen debido a la precipitación (granizo, nieve o lluvia en extremo), oleaje o fallas en alguna estructura hidráulica (presas hidroeléctricas, acueductos, entre otros). Provocan un incremento en el nivel de la superficie del agua de ríos, lagos, lagunas o el mar mismo, generando penetración de agua en sitios en los que usualmente no lo hay y, como consecuencia, genera daños en la población, agricultura, ganadería e infraestructura [1]. En el sur de México, estos eventos se pueden originar en las temporadas de lluvias, que comienzan a partir del mes de mayo y terminan en noviembre.

La percepción remota (PR) permite obtener información de un evento o fenómeno sin establecer contacto físico con él mediante un conjunto de herramientas y técnicas [8]. En la PR espacial se emplean sensores remotos para capturar información (imágenes) correspondientes a la interacción entre el flujo energético del Sol y la superficie terrestre, dichos sensores se encuentran en plataformas espaciales.

Las imágenes obtenidas por estos sensores poseen diferentes características que dependen de los sensores con las que son obtenidas, las cuales son: resolución espacial, que determina el área sobre la superficie terrestre que cubre cada píxel de la imagen; resolución espectral, indica el número y anchura de las regiones del espectro electromagnético captadas por el sensor remoto; resolución temporal, determina el tiempo en el que podemos adquirir información satelital del mismo lugar con el mismo satélite y resolución radiométrica, que indica la sensibilidad del sensor, es decir, la capacidad de discriminar entre pequeñas variaciones en la radiación que capta [7].

El Radar de Apertura Sintética (SAR, del inglés *Synthetic Aperture Radar*) es un sistema de visión lateral que se utiliza en la trayectoria de vuelo de la plataforma del sensor para simular una apertura de antena para generar imágenes satelitales de alta resolución [6].Esta tecnología se utiliza principalmente en países ecuatoriales para dar seguimiento a fenómenos dinámicos. Su principal objetivo es monitorear zonas terrestres independientemente de la condición climática prevalente al momento de la toma de la imagen.

Sentinel-1 es una constelación de dos satélites dirigidos por la Agencia Espacial Europea (ESA, del inglés *European Space Agency*). Tiene como objetivo dar continuidad a los datos del satélite ENVISAT Y ERS [4]. Opera en banda C (frecuencia central: 5.405 GHz) y posee una resolución temporal de 6 días. El objetivo principal de la misión son las aplicaciones para la vigilancia marina, el monitoreo de tierras y servicios de gestión de emergencia mediante el uso de la información que aportan las imágenes SAR que captan. Por otra parte,

Sentinel-2 [5] comprende una constelación de dos satélites de órbita polar colocados de forma sincrónica con el Sol. Su objetivo principal es monitorear las variaciones de la superficie terrestre ofreciendo imágenes de alta resolución de la misma zona cada 10 días. Los satélites incorporan un sensor óptico denominado "MultiSpectral Instrument"(MSI) que cuenta con 13 bandas espectrales; 4 de ellas cuentan con una resolución espacial de 10 metros, 6 con resolución de 20 metros y 3 con resolución de 60 metros [3].

El análisis y tratamiento los datos geoespaciales requieren de altos recursos informáticos, el cual se traduce en altos costos y limita la investigación en este campo. Google Earth Engine (GEE) ofrece una solución a esta limitante; es una plataforma basada en la nube que permite acceder y utilizar recursos informáticos de alto rendimiento dedicados al análisis de grandes colecciones de datos geoespaciales [9]. Además, la plataforma posee un robusto conjunto de datos que incluyen colección de imágenes satelitales Sentinel, Landsat, datos climáticos y de cobertura terrestre, entro otros [10]. La arquitectura de la plataforma permite la ejecución de algoritmos complejos en extensiones espaciales considerables de una manera rápida. Es una herramienta eficiente en diferentes aplicaciones geoespaciales como el mapeo de tierra de cultivo, detección de cambios en la cubierta terrestre, seguimiento y mapeo de desastres naturales.

En esta investigación se presenta una estrategia para la detección y seguimiento de zonas inundadas haciendo uso de imágenes satelitales Sentinel-1A SAR y Sentinel-2A en conjunto con la plataforma de datos geoespaciales Google Earth Engine. El presente documento está estructurado de la siguiente manera: en la Sección 2, se presentan los materiales y métodos utilizados para llevar a cabo la investigación; en la Sección 3, se presenta los resultados obtenidos; y finalmente en la Sección 4 se presentan las conclusiones obtenidas en el trabajo de investigación.

2. Materiales y métodos

La metodología propuesta para mapear las extensiones territoriales afectadas por la inundación se divide en dos etapas (ver Figura 1): 1) utilizando imágenes satelitales Sentinel-1A SAR, que consta de 4 subtareas y 2) utilizando imágenes satelitales Sentinel-2A, que consta de 4 subtareas y utiliza un método de aprendizaje automático. Adicionalmente, se utiliza una etapa adicional para recopilar los resultados que permiten su simplificación y evaluación. A continuación, se describe cada etapa.

2.1. Área de estudio

Tabasco se localiza al sureste de México y, en términos de superficie, ocupa el lugar 34 a escala nacional con una extensión territorial de 24661 km^2 que representa el 1.3 % del país. En la entidad se reconocen dos regiones y cinco subregiones político administrativas. La región Grijalva contiene la subregión

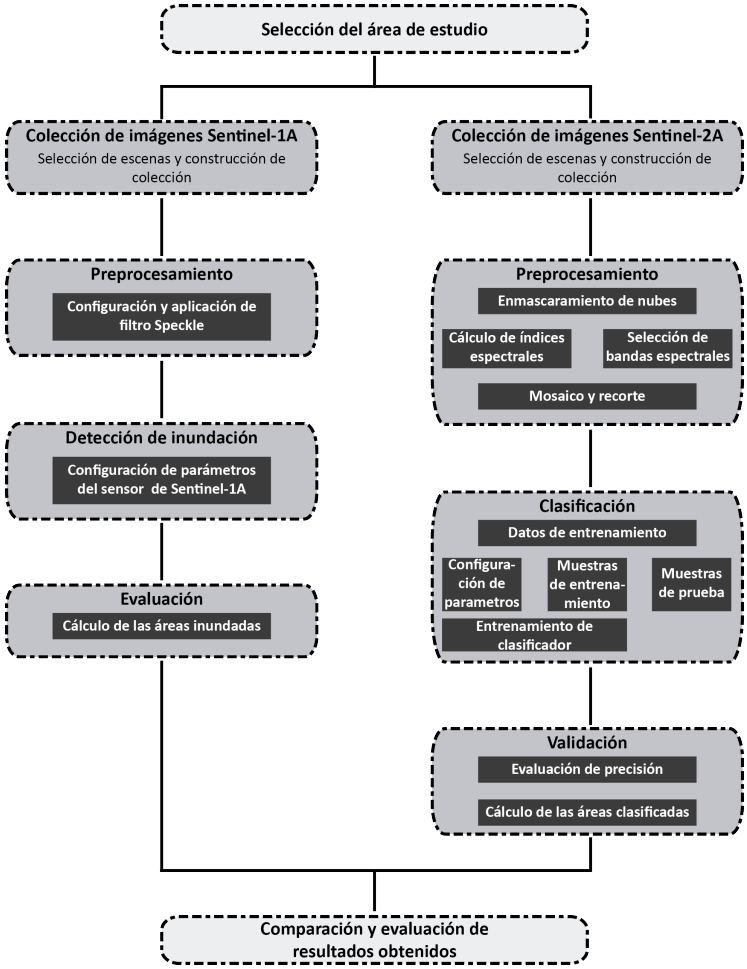


Fig. 1. Método implementado para la detección de zonas inundadas.

Chontalpa, Centro y Sierra. La región Usumacinta contiene dos subregiones: Pantanos y Ríos.

La subregión Ríos se localiza en la parte más oriental del estado, en los límites con el estado de Campeche y la República de Guatemala. Se llama así por la gran cantidad de ríos que la cruzan, entre ellos, el río Usumacinta, el más caudaloso del país y el río San Pedro Mártir.

Los municipios que integran esta subregión son: Tenosique, Emiliano Zapata y Balancán (ver Figura 2). Su superficie es de aproximadamente 6000 km^2, lo que representa el 24.67 % del total del estado; y su población, según cifras del INEGI era de 145 217 habitantes en el año 2010, es decir, el 6.24 % de la población total de la entidad.

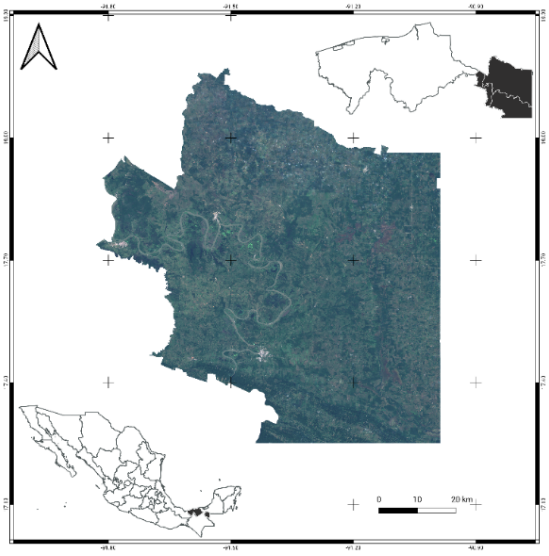


Fig. 2. Ubicación geográfica del área de estudio - Subregión Ríos de Tabasco.

2.2. Primera etapa metodológica Sentinel-1A (GRD)

Colección de imágenes Sentinel-1A (GRD). Para adquirir la colección de imágenes Sentinel-1A (S-1A) utilizadas en esta investigación, se determinaron dos series temporales; la primera temporada corresponde a los meses que comprende la temporada de seca en la región (marzo – mayo de 2020). La segunda temporada corresponde a los meses de octubre – noviembre del mismo año, periodo en el cuál ocurrieron las inundaciones en la región. Posteriormente se procedió a adquirir las imágenes S-1A en la plataforma *GEE Code*; para este fin, se utilizaron una serie de parámetros que sirvieron como filtros. La colección de imágenes fue filtrada para trabajar con la polarización *VH* que es utilizada para el mapeo de inundaciones. Para evitar señales de falso positivos, provocados por el ángulo de visión se utilizó la dirección de paso *descendente*. Finalmente, el área de estudio se delimito con ayuda de un archivo *shapefile* que contiene las coordenadas geográficas del área.

Preprocesamiento de colección de imágenes. En esta etapa se crearon mosaico de imágenes derivados de cada una de las colecciones de S-1A antes y después de las inundaciones de manera individual e independiente. Posteriormente, se recortaron las imágenes para delimitar el área de estudio, esto se logró gracias al archivo shapefile. Como último paso se aplicó el filtro *Speckle* para suavizar las imágenes y así reducir el efecto de sal – pimienta propia de las imágenes S-1A.

Detección de zonas inundadas. En esta etapa se calcula la diferencia entre las dos temporadas. El mosaico de la inundación es dividido por el mosaico antes de la inundación, como resultado se obtiene una capa ráster que contiene la información del grado de cambio perteneciente a cada píxel, el cual los valores altos indican cambios importantes mientras que los valores bajos indican pocos cambios.

Evaluación de resultados obtenidos. En esta etapa se calcula el área de extensión territorial ocupada por las inundaciones utilizando la capa ráster. Para ello se invoca a la polarización previamente definida y se multiplica por el área que representa cada píxel en metros cuadrados. Posteriormente se suman todos los píxeles de la inundación utilizando su resolución náutica de 10 metros. La información del área resultante se convierte en hectáreas.

Finalmente, como producto final se obtienen las imágenes correspondientes al antes y después de la inundación y la capa que discrimina los cuerpos de agua que tienen presencia mayor a 10 meses. Se resalta solo las áreas que presentan anomalías, es decir, las zonas inundadas además de mostrar las áreas en hectáreas ocupadas por las inundaciones.

2.3. Segunda etapa metodológica Sentinel-2A

Colección de imágenes Sentinel-2A. Para adquirir la colección de imágenes Sentinel-2A (S-2A), se determinaron dos series temporales, las cuales son las mismas utilizadas en la colección de imágenes S-1A. Esto se realizó con el objetivo de poder comparar las zonas afectadas por las inundaciones. Posteriormente, se adquirieron las imágenes S-2A en la plataforma GEE Code. Para ello, se utilizaron diferentes filtros para obtener imágenes de alta calidad. Se filtró mediante las temporadas antes mencionadas y el nivel de nubosidad menor al 40 %. El área de estudio se delimitó mediante el archivo shapefile que contiene las coordenadas geográficas del área de estudio.

Preprocesamiento de colección de imágenes. En esta etapa se aplica un enmascaramiento de nubes, a las colecciones de imágenes obtenidas en la etapa anterior, mediante la banda QA60 la cual guarda los bits que contienen las nubes y nubes grises en el bit 10 y 11, respectivamente. Con esta banda se permite eliminar los cúmulos de nubes que se encuentran presentes en las imágenes. Esta etapa se realiza para la obtención de imágenes limpias y evitar ruidos que pudieran interferir en los resultados. Posteriormente, las imágenes enmascaradas sustituyen a las originales en las colecciones.

Seguidamente, se calculan los índices espectrales para obtener información adicional acerca de la composición del suelo que permitirá obtener mejores resultados al utilizar métodos de aprendizaje automático. Para la vegetación se calculó el *NDVI*, *GNDVI*, *EVI* y *SAVI*. Para los cuerpos de agua se calculó el *NDWI* y *MNDWI*; estos índices se grabaron en forma de banda en cada una de sus imágenes correspondientes.

Finalmente se recortan las imágenes enmascaradas obtenidas en la etapa anterior ajustándolas a las coordenadas y contorno del área de estudio. De igual manera, se crean los mosaicos de imágenes y se realiza una reducción para compactar la información en una imagen por colección mediante el cálculo de su media aritmética.

Clasificación. En esta etapa se construye el conjunto de datos que se utilizara para entrena el método de aprendizaje supervisado. Los puntos de muestran se dividen en dos clases: 1) cuerpos de agua y 2) suelo – vegetación. La colección de datos recopilados se dividió en dos colecciones, 70 % para entrenamiento y 30 % para validación. El método de aprendizaje utilizado es la Maquina de Soporte de Vectores (SVM) [2]. Dicho método ha sido utilizado en distintas investigaciones con resultados satisfactorios. En este método se configuró con un kernel de tipo función de base radial con una gamma de 0.7 y un costo de 30; los parámetros se obtuvieron mediante diversas pruebas para encontrar los valores óptimos. El método SVM se entrenó utilizando como base el mosaico de imágenes de la temporada seca. Con el método SVM previamente entrenado se clasificaron los mosaicos de imágenes de la otra temporada. Como resultado se obtiene una nueva capa ráster clasificada.

Validación. En esta etapa se evalúa el desempeño del algoritmo SVM. Se utilizó la validación cruzada que evalúa los resultados obtenidos por el método. Se valida con el conjunto de datos reservado para este objetivo. Con ello se obtiene la matriz de confusión, Índice Kappa, precisión general del entrenamiento, exactitud del usuario y productor. Si los valores del índice Kappa y precisión general se acercan a "0", los resultados serán menos confiables; si se acercan a "1", los resultados son más confiables.

Seguidamente es necesario obtener el área que ocupa cada uno de los píxeles y convertirlos en hectáreas. Las imágenes ráster obtenidas por el método SVM se le renombran las clases obtenidas. Posteriormente, se suman todos los pixeles de cada clase para obtener el área total ocupada por cada una.

Finalmente, como producto final se obtienen los ráster S-2A clasificados por el método SVM de cada una de las temporadas el cual agrega una paleta de colores para diferenciar cada clase. Además, se agrega la información que cada clase ocupa en cada una de las imágenes clasificadas en hectáreas.

2.4. Comparación y evaluación de resultados obtenidos

En esta última etapa se comparan los resultados y la detección de inundación ante las imágenes S-1A GRD y S-2A (clasificados por el método SVM). Este paso se realiza para identificar errores en la predicción con la utilización de ambos datasets de imágenes y discriminar los que presenten anomalías.

3. Resultados

Para el estudio se obtuvieron dos conjuntos de imágenes: 1) antes de la inundación, marzo-mayo 2020 y 2) durante el evento de la inundación, octubre-noviembre 2020. Los resultados obtenidos se dividieron en dos secciones: 1) utilizando imágenes Sentinel-1A y 2) utilizando imágenes Sentinel-2A.

3.1. Sentinel-1A (GRD)

Aplicación de filtro Speckle. La información de las imágenes Sentinel-1A SAR que GEE provee se encuentran previamente preprocesadas, es decir, se obtienen con corrección y calibración radiométrica al igual que el ruido térmico es eliminado. Se aplica un algoritmo de suavizado para reducir el efecto moteado o Speckle, esto con el objetivo de obtener imágenes de mayor calidad. En la Figura 3 muestran nuestra área de estudio antes y después de aplicar el suavizado. Estas imágenes fueron capturadas durante la temporada seca de 2020.

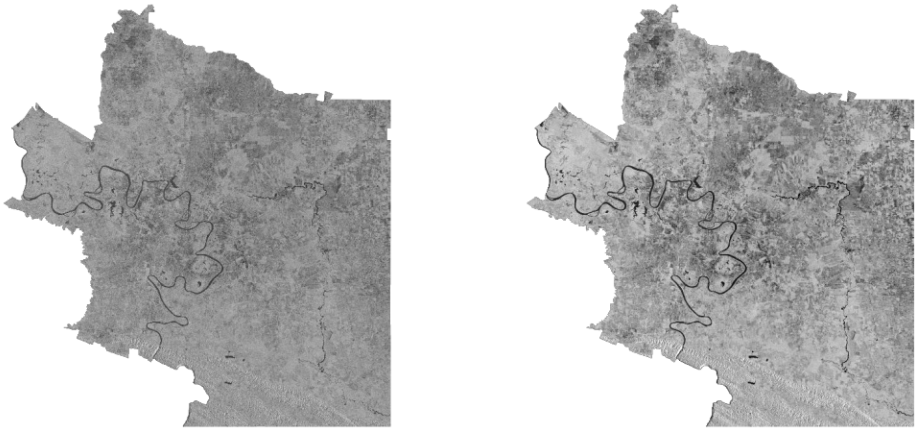


Fig. 3. Izquierda: Área de estudio sin filtro Speckle. Derecha: Área de estudio con filtro Speckle.

Detección de cambios. Para este apartado se utilizó un enfoque de detección de cambios, donde el mosaico de imágenes SAR, posterior a la inundación, se divide entre el mosaico de imágenes SAR anterior a la inundación. Esto da como resultado una nueva imagen que muestra los cambios presentados por pixel. Los pixeles brillantes indican un cambio alto, mientras que los pixeles más oscuros indican pequeños cambios en los mosaicos de imágenes. Para lograr esta detección de cambios se definió un umbral de 1.25, en el cual todos los valores superiores a 1.25 se le asigna un valor de 1 (inundado) y a todos los valores

inferiores a 1.25 se les asigna un valor de 0 (no inundado). En la Figura 4 se presenta la imagen obtenida con la detección de áreas inundadas.

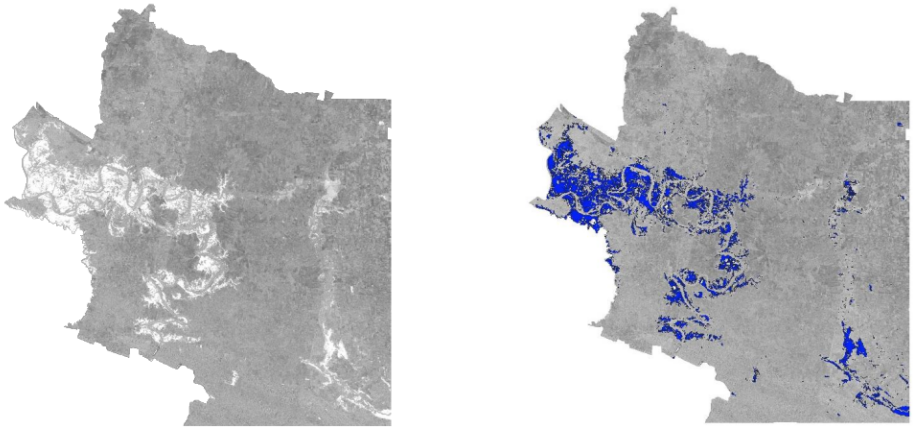


Fig. 4. Izquierda: capa de diferencia, las áreas brillantes indican un cambio alto, las áreas oscuras poco cambio. Derecha: capa de extensión de inundación resultante aplicando un umbral de 1,25.

Cálculo y refinamiento de la extensión territorial inundada. Para obtener resultados satisfactorios y precisos se utilizaron varios conjuntos de datos adicionales para eliminar falsos positivos dentro de la capa de extensión de inundación. El conjunto de datos del JCR Global Surface Water[1] se utilizó para enmascarar todas las áreas cubiertas por agua durante más de 10 meses al año. Este conjunto de datos tiene una resolución espacial de 30 metros y su última actualización fue en el año de 2018.

Posteriormente, se llevó a cabo la implementación de un algoritmo computacional matemático para obtener la extensión de la inundación. El algoritmo crea una nueva capa calculando el área en m^2para cada píxel; al sumar todos los píxeles, la información del área se deriva y se convierte en hectáreas. En la Figura 5 se muestra la imagen SAR obtenida de la inundación presentada en noviembre de 2020; de acuerdo a los datos obtenidos, el área afectada fue de 39,815 hectáreas de la subregión Ríos del estado de Tabasco.

3.2. Sentinel-2A

Mapeo de cuerpos de agua y zonas inundadas. Las imágenes para este estudio, fueron adquiridas por el satélite Sentinel-2A y procesadas en la plataforma GEE para detectar y mapear los cuerpos de agua e inundación a través de

[1] https://developers.google.com/earth-engine/datasets/catalog/JRC_GSW1_1_GlobalSurfaceWater

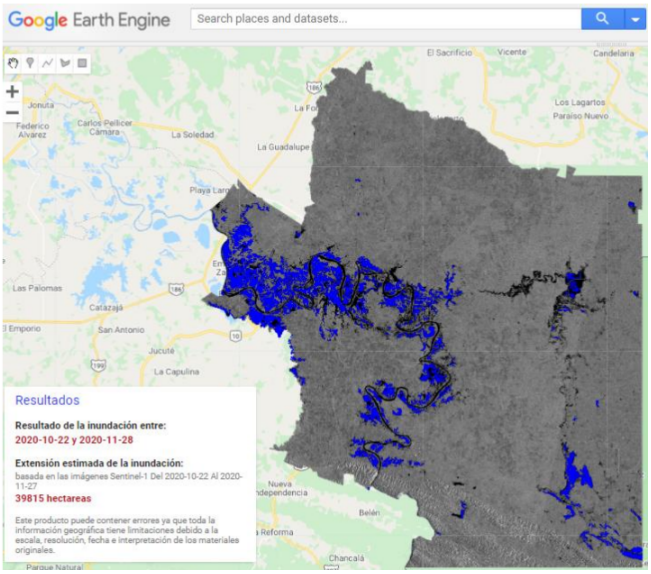


Fig. 5. Resultados obtenidos de las inundaciones de la subregión Ríos de Tabasco visto desde el visor de mapas de GEE.

clasificación supervisada con el algoritmo computacional SVM. Estas imágenes fueron clasificadas en dos clases: 1) cuerpos de agua y 2) suelo – vegetación. En la Figura 6 se presenta el mapeo e identificación de cuerpos de agua antes de la inundación. Asimismo, se muestran las zonas afectadas por la inundación de noviembre de 2020.

Precisión de clasificación. Para evaluar la exactitud de los resultados obtenidos por el algoritmo SVM, se utilizaron un total de 897 píxeles de los cuales: 464 corresponden para la clase cuerpos de agua y 433 para suelo – vegetación. Estos datos fueron usados para calcular la matriz de confusión. la precisión general de la clasificación e Índice Kappa. En la Tabla 1, se presenta la precisión obtenida por el algoritmo en los dos momentos temporales analizados.

Tabla 1. Resultados de evaluación de exactitud del método de clasificación (algoritmo SVM).

Temporada	**Evaluación de precisión**	
	Precisión general	**Índice KAPPA**
Seca	1	1
Inundación	0.97	0.95

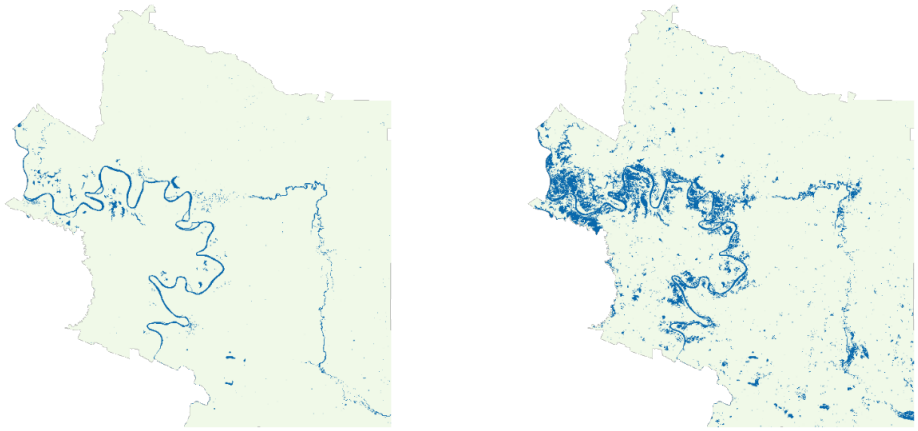


Fig. 6. Mapeo e identificación de cuerpos de agua e inundación en la subregión Ríos de Tabasco. Izquierda: cuerpos de agua en temporada seca 2020. Derecha: cuerpos de agua y áreas inundadas.

Estimación de extensión territorial. Detectar las variaciones en la extensión territorial de los cuerpos de agua, en tiempo de inundaciones, es un tema de importancia para diversas organizaciones encargadas de prevenir y actuar ante desastres naturales. Bajo este contexto, se realizó una estimación de la extensión territorial, ocupada por cada una de las clases involucradas en la clasificación, para visualizar las variaciones e impacto que causaron las inundaciones en nuestra área de estudio. Se implementó un algoritmo computacional matemático para obtener la extensión de la inundación; dicho algoritmo crea una nueva capa calculando el área en m^2 para cada píxel. Al sumar todos los píxeles, la información del área se deriva y se convierte en hectáreas.

En la Tabla 2 se muestran las variaciones de extensión territorial en términos de hectáreas que han presentado cada una de las clases involucradas en este proceso. Los datos obtenidos demuestran que la clase cuerpos de agua aumentó 473.71 % su extensión territorial durante las inundaciones (respecto a la temporada seca del 2020). La clase suelo – vegetación disminuyó 10.86 % su extensión territorial debido a las inundaciones que se presentaron en la temporada en estudio.

Tabla 2. Extensión territorial por temporadas de estudio de las clases clasificadas por el algoritmo SVM.

Temporada	**Extensión territorial por clase (Hectáreas)**	
	Cuerpos de agua	**Suelo - vegetación**
Seca	13225.939	591904.59
Inundación	75878.27	527617.91

4. Conclusiones

En este trabajo de investigación se han obtenido la información de las inundaciones ocurridas en la subregión Ríos del estado de Tabasco durante los meses de octubre – noviembre de 2020. Para ello, se implementó algoritmos computacionales en imágenes satelitales Sentinel-1A y Sentinel-2A. Los resultados presentados muestran que la información obtenida de imágenes SAR son de gran importancia para el monitoreo de emergencias y desastres naturales. Estas imágenes se pueden obtener bajo condiciones adversas climáticas o atmosféricas como lo son lluvia, llovizna, nubosidad, entre otras. Asimismo, las imágenes Sentinel-2A son útiles para obtener información acerca de la dinámica terrestre, pero pocas efectivas ante condiciones adversas porque la presencia de nubes o ruidos afectan los resultados.

La combinación de estas tecnologías y herramientas permitieron determinar zonas inundadas y obtener un estimado de la extensión territorial afectadas por las inundaciones. Este estudio podría ser utilizada por instituciones como el Instituto Nacional de Estadística y Geografía (INEGI), Centro Nacional de Prevención de Desastres (CENAPRED) y gobiernos estatales para dar seguimiento a fenómenos similares en el futuro.

Referencias

1. Centro Nacional de Prevención de Desastres: ¿Qué es una inundación?, pp. 1–5. 1 edn. (2012), https://www1.cenapred.unam.mx/DIR_SERVICIOS_TECNICOS/SANI/Entidades%20Federativas/Recursos/Inundaciones/190502_RI_Folleto%20de%20inundaci%C3%B3n_mod.pdf
2. Cortes, C., Vapnik, V.: Support-vector networks. Machine Learning 20(3), 273–297 (1995), https://link.springer.com/content/pdf/10.1007/BF00994018.pdf
3. Delegido, J., Tenjo, C., Ruiz-Verdu, A., Pereira-Sandoval, M., Pasqualotto, N., Gibaja, G., Verrelst, J., Peña, R., Urrego, P., Borràs, J., Sanchis Muñoz, J., Pezzola, A., Mosquera, Z., Quinto, Z., Gómez, J., Moreno, J.: Aplicaciones de sentinel-2 a estudios de vegetación y calidad de aguas continentales (2016), https://www.researchgate.net/publication/311572244_Aplicaciones_de_Sentinel-2_a_estudios_de_vegetacion_y_calidad_de_aguas_continentales
4. European Space Agency: Sentinel-1 (2014), https://sentinel.esa.int/web/sentinel/missions/sentinel-1
5. European Space Agency: Sentinel-2 (2015), https://sentinel.esa.int/web/sentinel/missions/sentinel-2
6. Fernández-Ordoñez, Y., Soria-Ruiz, J., Leblon, B., Macedo, A., Elva, M., Ramírez Guzmán, M.E., Escalona-Maurice, M.: Imágenes de radar para estudios territoriales, caso: inundaciones en tabasco con el uso de imágenes sar sentinel-1a y radarsat-2 11, 4–21 (2020), https://www.researchgate.net/publication/340333844_Imagenes_de_radar_para_estudios_territoriales_caso_inundaciones_en_Tabasco_con_el_uso_de_imagenes_SAR_Sentinel-1A_y_Radarsat-2
7. Labrador García, M., Brondo, J., Arbelo, M.: Satélites de teledetección para la gestión del territorio (2012), https://www.researchgate.net/

publication259230060_Satelites_de_teledeteccion_para_la_gestion_del_territorio

8. Lillesand, T., Kiefer, R., Chipman, J.: Remote Sensing and Image Interpretation, vol. 146. 5 edn. (2004), https://www.researchgate.net/publication/235863921_Remote_Sensing_and_Image_Interpretation_Fifth_Edition
9. Liss, B., Howland, M.D., Levy, T.E.: Testing google earth engine for the automatic identification and vectorization of archaeological features: A case study from faynan, jordan. Journal of Archaeological Science: Reports 15, 299–304 (2017), https://www.sciencedirect.com/science/article/pii/S2352409X16308082
10. Xiong, J., Thenkabail, P.S., Gumma, M.K., Teluguntla, P., Poehnelt, J., Congalton, R.G., Yadav, K., Thau, D.: Automated cropland mapping of continental africa using google earth engine cloud computing. ISPRS Journal of Photogrammetry and Remote Sensing 126, 225–244 (2017), https://www.sciencedirect.com/science/article/pii/S0924271616301575

Electronic edition
Available online: http://www.rcs.cic.ipn.mx

ISSN: 1870-4069
http://rcs.cic.ipn.mx

Centro de Investigación
en Computación