RESUMEN TRABAJO DE GRADO

AUTOR(ES):

NOMBRE(S): GHIORDY FERNEY APELLIDOS: CONTRERAS CONTRERAS

NOMBRE(S): APELLIDOS:

FACULTAD: INGENIERÍA

PLAN DE ESTUDIOS: INGENIERÍA ELECTRÓNICA

DIRECTOR:

NOMBRE(S): BYRON APELLIDOS: MEDINA DELGADO

CODIRECTOR:

NOMBRE(S): BRAYAN RENE APELLIDOS: ACEVEDO JAIMES

TÍTULO DEL TRABAJO (TESIS): CLUSTER CV2: UN ABORDAJE DE VISIÓN COMPUTACIONAL PARA IDENTIFICACIÓN ESPACIAL DE AGRUPAMIENTOS DE DATOS

RESUMEN

El proyecto tiene como finalidad desenvolver una metodología de agrupamientos de datos para muestras con problemas de superposición entre grupos, usando las herramientas de visión computacional y aprendizaje de máquina, donde la mayor parte de las bases de datos reales presentan este inconveniente para la correcta identificación de los grupos y generando pérdida de información. Usando la metodología de desarrollo de aplicaciones móviles en modo iterativo sobre las etapas de: análisis, diseño, desarrollo, pruebas y entrega, estructurando la metodología eficazmente y evaluando los requerimientos técnicos, funcionales y de accesibilidad para el software final. Por consiguiente, el documento describe el desarrollo del proyecto en 5 capítulos, donde el primer capítulo describe la propuesta metodológica planteada, metas y resultados a los cuales se han orientado el proyecto; continuando con el capítulo 2, se ha abordado la revisión literaria de los trabajos en el área, extrayendo los métodos de procesos y aplicando estos en el capítulo 3 del presente documento. Finalmente, en los capítulos 4 y 5 del documento se presentan los resultados obtenidos, la discusión y el análisis de estos, identificando los factores de error, la relevancia del trabajo para el área abordada y el aporte de este en el área de Ingeniería Electrónica.

PALABRAS CLAVE: Tratamiento de datos, visión computacional, machine learning.

PÁGINAS: 148 PLANOS: ____ ILUSTRACIONES: ____ CD ROOM: 1

CLUSTER CV2: UN ABORDAJE DE VISIÓN COMPUTACIONAL PARA IDENTIFICACIÓN ESPACIAL DE AGRUPAMIENTOS DE DATOS

GHIORDY FERNEY CONTRERAS CONTRERAS

UNIVERSIDAD FRANCISCO DE PAULA SANTANDER

FACULTAD DE INGENIERÍA

PLAN DE ESTUDIOS DE INGENIERÍA ELECTRÓNICA

CÚCUTA

2020

CLUSTER CV2: UN ABORDAJE DE VISIÓN COMPUTACIONAL PARA IDENTIFICACIÓN ESPACIAL DE AGRUPAMIENTOS DE DATOS

GHIORDY FERNEY CONTRERAS CONTRERAS

Trabajo de grado presentado para optar por el título de Ingeniero Electrónico

DIRECTOR: PhD. BYRON MEDINA DELGADO

CODIRECTIOR: PhD. (c) BRAYAN RENE ACEVEDO JAIMES

UNIVERSIDAD FRANCISCO DE PAULA SANTANDER

FACULTAD DE INGENIERÍA

PLAN DE ESTUDIOS DE INGENIERÍA ELECTRÓNICA

CÚCUTA

2020

Universidad Francisco de Paula Santander
Vigilada Mineducación
NIT. 890500622 - 6

ACTA DE SUSTENTACIÓN DE UN TRABAJO DE GRADO

Fecha:	CÚCUTA, 16 DE DICIEMBRE DE 2019
Hora:	10:00
Lugar:	LABORATORIO EMPRESARIAL, LE101
Plan de Estudios:	INGENIERÍA ELECTRÓNICA
Título de la Tesis:	"CLUSTER CV2: UN ABORDAJE DE VISIÓN COMPUTACIONAL PARA IDENTIFICACIÓN ESPACIAL DE AGRUPAMIENTOS DE DATOS"
Jurados:	IE MSc. ANDRÉS EDUARDO PAEZ PEÑA IE ESp. SERGIO IVÁN QUINTERO AYALA
Director: Codirector:	IE PhD BYRON MEDINA DELGADO PhD (c) BRAYAN RENE ACEVEDO JAIMES

Nombre del Estudiante	Código	Calificación
GHIORDY FERNEY CONTRERAS CONTRERAS	1161146	CINCO, CERO (5,0)

LAUREADA

ANDRÉS EDUARDO PAEZ PEÑA

SERGIO IVÁN QUINTERO AYALA

DINAEL GUEVARA IBARRA, IE PhD
Coordinador Comité Curricular
Ingeniería Electrónica

CARTA DE AUTORIZACIÓN DE LOS AUTORES PARA LA CONSULTA, LA REPRODUCCIÓN PARCIAL O TOTAL Y LA PUBLICACIÓN ELECTRÓNICA DEL TEXTO COMPLETO

Cúcuta,

Señores
BIBLIOTECA EDUARDO COTE LAMUS
Ciudad

Cordial saludo:

Ghiordy Ferney Contreras Contreras, identificado(s) con la C.C. Nº 1090503143, autor(es) de la tesis y/o trabajo de grado titulado CLUSTER CV2: UN ABORDAJE DE VISIÓN COMPUTACIONAL PARA IDENTIFICACIÓN ESPACIAL DE AGRUPAMIENTOS DE DATOS presentado y aprobado en el año 2019 como requisito para optar al título de Ingeniero Electrónico; autorizo(amos) a la biblioteca de la Universidad Francisco de Paula Santander, Eduardo Cote Lamus, para que con fines académicos, muestre a la comunidad en general a la producción intelectual de esta institución educativa, a través de la visibilidad de su contenido de la siguiente manera:

- los usuarios pueden consultar el contenido de este trabajo de grado en la página web de la Biblioteca Eduardo Cote Lamus y en las redes de información del país y el exterior, con las cuales tenga convenio la Universidad Francisco de Paula Santander.

- Permita la consulta, la reproducción, a los usuarios interesados en el contenido de este trabajo, para todos los usos que tengan finalidad académica, ya sea en formato CD-ROM o digital desde Internet, Intranet etc.; y en general para cualquier formato conocido o por conocer.

Lo anterior, de conformidad con lo establecido en el artículo 30 de la ley 1982 y el artículo 11 de la decisión andina 351 de 1993, que establece que "**los derechos morales del trabajo son propiedad de los autores**", los cuales son irrenunciables, imprescriptibles, inembargables e inalienables.

Ghiordy Contreras 1.090.503.143

FIRMA Y CEDULA

Dedicatoria

Este trabajo es dedicado a:

- Toda mi familia en especial a mi mamá Gloria María Contreras Contreras, a mi tía Luz Marina Contreras Contreras, mis hermanas Karen Sofia Galvis Contreras, Diana Carolina Galvis Contreras, y a mi novia Ingrid Vanessa Dolores Chávez, quienes son los motores de mi vida, gracias a ellas, por todo su amor, motivación y su apoyo incondicional.
- Mis abuelas Angela Contreras y Cecilia Rolón, que siempre me han cuidado y estado conmigo en cada día de mi vida.
- Todos mis compañeros de estudio durante mi formación profesional, por su apoyo, motivación y colaboración.

Todo este trabajo ha sido posible gracias a ellas y ellos.

Ghiordy Ferney Contreras Contreras

Agradecimientos

El autor expresa sus agradecimientos a:

- Mis orientadores el Doctor Byron Medina Delgado, el candidato a Doctor Brayan Rene Acevedo Jaimes y el Doctor Dinael Guevara Ibarra, por su confianza, enseñanza, colaboración, motivación, y apoyo durante mi proceso de formación profesional y aprendizaje.
- El Doctor Héctor Jaime Dulcé Moreno, quien desde hace cuatro años me ha orientado en el desarrollo de mis habilidades científicas, intuitivas y cognitivas para la ejecución de proyectos de investigación en ciencias exactas y aplicadas.
- Todos los miembros del Grupo de Investigación y Desarrollo en Electrónica y Telecomunicaciones (GIDET), en especial al Ingeniero Pedro Beltrán, al Ingeniero Jesús Álvarez y al compañero Juan Galvis, por sus orientaciones y aclaraciones acerca del desarrollo de un trabajo de investigación.
- Mi madre, por toda su comprensión y sacrificio hecho para que lograse tener todo lo que requería en mi formación.

Contenido General

Lista de tablas

pág.

Lista de figuras

pág.

Lista de anexos

pág.

Resumen

En este documento se presenta el trabajo completo desarrollado para el proyecto Cluster CV2, con 5 capítulos de desarrollo divididos en: Descripción del problema, Marco referencial, Metodología, Resultados y Conclusiones.

En el capítulo 1, se encuentra la contextualización básica, donde se describe claramente la problemática a tratar, su justificación y los objetivos del trabajo, así mismos detalles metodológicos como el tipo, delimitaciones, limitaciones y resultados o impacto esperado.

En el capítulo 2, se amplía los conceptos, usando referentes literarios y artículos tanto de revistas especializadas como de resumen de conferencia, adicionalmente al final del capítulo de encuentra la normativa local e internacional que enmarca el desarrollo del proyecto.

En el capítulo 3, se describe el desarrollo metodológico aplicado etapa a etapa, usando las herramientas dispuestas en el marco referencial con el objeto lograr cada uno de los objetivos propuestos.

En el capítulo 4, se describe cada uno de los resultados alcanzados, como los validados por métricas de agrupamiento y su principal diferenciador de otros resultados. Por otro lado, se aborda los abordajes con datos reales y los resultados encontrados en la prueba y error. Al final del capítulo se encuentra una breve relación de como estos han sido divulgados públicamente. En el capítulo 5, se encuentra la concesión de resultados, algunas previamente publicadas y como se lograría mejorar el abordaje.

Abstract

This document presents the complete work developed for the Cluster CV2 project, with 5 development chapters divided into: Description of the problem, Reference framework, Methodology, Results and Conclusions.

In chapter 1, there is the basic contextualization, which clearly describes the problem to be treated, its justification and the objectives of the work, as well as methodological details such as the type, delimitations, limitations and results or expected impact.

In Chapter 2, the concepts are expanded, using literary references and articles from both specialized journals and conference summary, additionally at the end of the chapter, it finds local and international regulations that frame the development of the project.

Chapter 3 describes the methodological development applied step by step, using the tools set out in the referential framework in order to achieve each of the proposed objectives.

In Chapter 4, each of the results achieved is described, such as those validated by grouping metrics and their main differentiator from other results. On the other hand, the approaches with real data and the results found in the trial and error are addressed. At the end of the chapter there is a brief relationship of how these have been publicly disclosed. In chapter 5, there is the granting of results, some previously published and how the approach would be improved.

Introducción

Un dato es básicamente cualquier tipo de información adquirible y cuantificable, como lo son las medidas de un sensor, pixeles de una imagen, medidas de un radar, palabras de un texto, tweets, puntos de GPS, medidas de temperatura, valores de predicción en la bolsa de valores, etc. Sin embargo, con el desarrollo tecnológico para la adquisición de datos han aumentado considerablemente la cantidad de información en las bases de datos, llevándonos a la era del Big data, donde el análisis de datos va requiriendo de herramientas cada vez más sofisticadas y de aprendizaje automático para la extracción de información relevante en los conjuntos de datos con gran cantidad de dimensiones, superposición de muestras o desconocimiento de los grupos de muestras dentro del conjunto de datos (Davies, 2012).

En análisis de datos, el agrupamiento es considerado como la clasificación no supervisada de un conjunto de datos, donde a través de métricas o parámetros, es posible identificar similaridad o disimilaridad entre las muestras pertenecientes a un conjunto de datos (Jaimes, Castro, Torres, Silva, & Braga, 2017). Un ejemplo práctico del agrupamiento de datos ocurre en el diagnóstico de enfermedades, donde un grupo de personas presentan la misma enfermedad al tener similaridad de síntomas entre ellas, y en el caso de que alguna persona externa presente los mismos síntomas, esta procederá a ingresar al grupo. Debido a este tipo de sucesos, el agrupamiento de datos y su análisis presenta un amplio rango de aplicaciones como la delimitación de especies biológicas, compresión de datos, clasificación de enfermedades (fisiológicas o mentales), entre otras (Hennig, Meila, Murtagh, & Rocci, 2015). Por esto, se encuentra en la literatura variedad de algoritmos para el agrupamiento de datos (Mwangi, Soares, & Hasan, 2014), donde los más populares son: k-means (Tafsast, Hadjili, Bouakaz, & Benoudjit, 2017), usado para enfermedades (Srinivas & Rao, 2018) y pronóstico ambiental (Zscheischler,

Mahecha, & Harmeling, 2012); fuzzy c-means, usada para datos con superposición (Zhang, Wang, & Zhang, 2007) y en técnicas de segmentación (Alban, et al., 2018); jerárquico, aplicado como agrupamientos multietapa en algoritmos de filtrado para comunicaciones de luz visible (Mitra & Bhatia, 2017); espectral, cada muestra tiene un nivel de pertenencia a cada grupos (Silva, y otros, 2017), es decir, se presenta relación entre los diferentes grupos (Vora & Raman, 2018).

Por otro lado, la visión computacional sumergida dentro del procesamiento digital de señales es abordada como las técnicas que permiten incluir dentro de una maquina el sentido de la visión como si fuese un ser humano. Para lograr esto se cuenta con variedad de operadores morfológicos basados en conceptos matemáticos como la transformada discreta de fourier, rotacional, divergente, límites, entre otros. La visión computacional permite a la maquina procesar las imágenes a un nivel intuitivo, donde de una imagen se logra extraer información basada en los criterios de contexto, cognición y adaptabilidad descritos en el modelo de procesamiento de la máquina. En este contexto se propone abordar herramientas de procesamiento digital de imágenes sobre una matriz de similaridad proyectada como imagen dentro de un sistema computacional, donde esta contiene información relevante con respecto a que grupos debe ser atribuida cierta muestra y como está se relaciona con el resto de las muestras contenidas en el conjunto de datos.

1. Descripción del Problema

Inicialmente, se ha planteado una problemática que se ha evidenciado por varios autores en la literatura, siendo comprendida bajo ciertas características y descrita de la siguiente manera:

1.1. Planteamiento del problema

En los repositorios de Machine Learning como (Dheeru & Karra Taniskidou, 2017) se pueden encontrar variedad de bases de datos para diferentes áreas en general, estas bases de datos son usados ampliamente por científicos, ingenieros, analistas, entre otros (Davies, 2012). En tanto, la mayoría de las bases de datos no presentan información a priorí para definir como clasificar o agrupar los datos obtenidos según las características de los datos y la mayoría de estos presentan superposición entre ellos, como se presenta en la Figura 1.

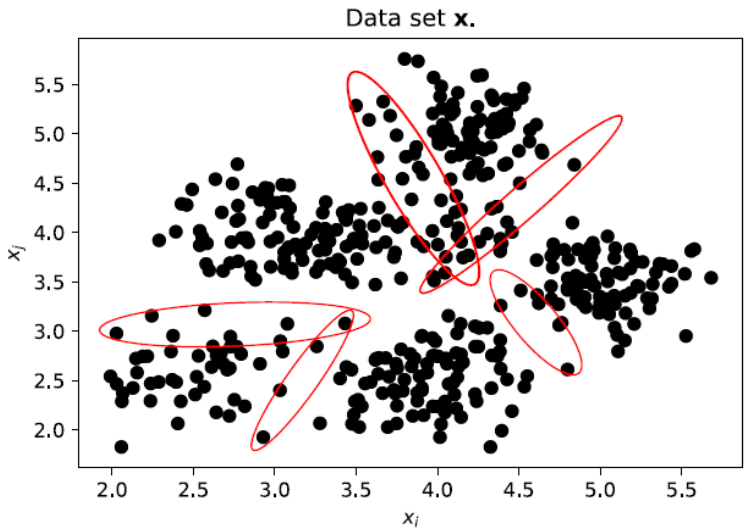


Figura 1. Conjunto de datos con superposición.

El problema de la superposición ilustrado en la Figura 1, para la cual se cuenta con un conjunto de datos sintéticos generado a través distribuciones gaussianas donde la desviación estándar (s_d) representa el grado de superposición, para el ejemplo este grado es definido en 0,3 con 900 muestras en el conjunto de datos. Este tipo de datos tienden a generar agrupamientos indeseables, debido a que la mayoría de los algoritmos de agrupamiento de datos en la literatura,

son de naturaleza particional, es decir, los grupos no presentan relación entre ellos $C_j \cap C_k = 0$. Por otro lado, técnicas de agrupamiento jerárquico abordan el paradigma de que cada muestra tiene un nivel de pertenencia a cada grupo $C = U_{j=1}^{m} C_j$, es decir, existe relación entre grupos de datos dentro del conjunto de datos $C_j \cap C_k \neq 0$ para muestras de diferentes grupos (Jaimes, Castro, Torres, Silva, & Braga, 2017). La ecuación (1) define el grado de superposición entre los grupos de datos contenidos en conjunto de datos, donde N es la cantidad de datos, k es el número de grupos contenidos, z_i es el centroide para cada grupo C_i, i es una contante definida como $i = 1,2,3, \dots, k$ y j es una constante definida como $j = i + 1, i + 2, \dots, k$.

$$\boldsymbol{val} = \frac{\boldsymbol{d_{inter-samples}}}{\boldsymbol{d_{inter-clusters}}} = \frac{\frac{\mathbf{1}}{\boldsymbol{N}} \sum_{i=1}^{k} \sum_{x \varepsilon C_i} \|\boldsymbol{x} - \boldsymbol{z_i}\|^2}{\boldsymbol{min_{i,j}} \left(\|\boldsymbol{z_i} - \boldsymbol{z_j}\|^2 \right)} \tag{1}$$

1.2. Justificación

Esta investigación propone un tratamiento espacial al agrupamiento de datos con superposición si eliminar información, corrigiendo la superposición y generar datos más interpretables para el usuario que la use, tal como es el propósito del agrupamiento (Vora & Raman, 2018).

Al no contar con una herramienta para datos superpuestos, se generan agrupamientos indeseables y con precisión media como lo han expuesto en los trabajos (Kwon, y otros, 2018), sin embargo, existen varias técnicas para datos superpuestos como lo son el agrupamiento jerárquico (Hennig, Meila, Murtagh, & Rocci, 2015), el algoritmo OClustR (Pérez-Suárez, Martínez-Trinidad, Carrasco-Ochoa, & Medina-Pagola, 2013), el algoritmo fuzzy (Zhang, Wang, & Zhang, 2007), entre otros, pero pocos han trabajado la corrección de la superposición de los datos como el Cluster CV original (Jaimes, Castro, Torres, Silva, & Braga, 2017).

Los resultados del Cluster CV2 apuntan a obtener una clasificación más precisa sobre datos con superposición como lo presentan los trabajos (Ng, Jordan, & Weiss, 2002) y corregir dicha superposición dentro de otro espacio lineal, en el cual el usuario al ejecutar la aplicación visualizando los grupos sin dificultad, para su posterior análisis de los datos en esta era del Big Data.

Doctores y astronautas agrupan las propiedades de los fenómenos como tumores (Bi, Sun, & Xu, 2017) o galaxias respectivamente, para el análisis de estos y también se puede mencionar como las empresas clasifican a sus clientes según sus gustos, necesidades, entre otras características (Hennig, Meila, Murtagh, & Rocci, 2015). Así mismo, la ciencia va creciendo cada día más, donde el uso de herramientas de software va facilitando y abriendo camino para un mayor crecimiento en interpretación de los fenómenos, en el caso del agrupamiento de datos se encuentran trabajos para comunicaciones (Khanmohammadi, Adibeig, & Shanehbandy, 2017), pronóstico climático (Grabner, Bregar, Ivanjko, & Valencic, 2017), enfermedades psiquiátricas (Zhao, y otros, 2018), reconocimiento de voz (Rath, 2017), etc.

1.2.1. Beneficios tecnológicos.

La visión computacional da a las máquinas el sentido del ser humano de percibir el mundo real, donde los principales factores de la visión: son la intensidad de luz, el color y el contexto. Por lo tanto, esta permite darle a la computadora la perspectiva que tendría una persona desde el punto de vista de la visión.

Las técnicas Machine learning permiten aprendizaje automático desde los datos (como sus siglas lo describen), donde a través de parámetros matemáticos y estadísticos es posible identificar situaciones o modificar los parámetros de manera automática.

El Software de código abierto permite reducir costos de producción como de implementación, donde el costo de producción en masa de un equipo o un software requiere de proyectos a largo plazo y con financiaciones elevadas.

1.2.2. Beneficios científicos.

El agrupamiento de datos para Ingeniería quiere despertar el interés en realizar mejores prácticas estadísticas de los resultados, donde generalmente se usan solo conceptos básicos de estadística descriptiva o inferencial, dejando de lado un análisis profundo de los resultados.

La extracción de características de manera automática permite dar a investigadores maneras intuitivas de evaluar sus datos, donde para investigaciones exploratorias o altamente variantes no existen parámetros certeros sobre los fenómenos que traten.

Los datos masivos generalmente tienen información intrínseca, donde al abordarlos como un solo conjunto no logra expresar comportamientos atípicos, de esto el agrupamiento de datos puede segmentar estos en subconjuntos similares y con una menor cantidad de muestras.

1.3. Alcance

Esta investigación es de tipo exploratoria cuantitativa donde se aborda un tema poco trabajado en la literatura, como lo es el agrupamiento de datos con superposición. Así, la investigación solo llega hasta aspectos teóricos, donde sus resultados no son aplicados tangiblemente, pero son evaluados con otros trabajos en la literatura y abordando el paradigma de sistemas no paramétricos.

1.3.1. Resultados esperados.

Abordar sistemas no paramétricos para la identificación de patrones y extracción grupos de conjuntos de datos sin información a priori para la clasificación.

Elevar la precisión y exactitud en la obtención de grupos con ciertos grados de superposición entre muestras de diferentes grupos.

Aplicar el procesamiento digital de señales para la identificación de patrones y características en conjuntos de datos.

Generar en los científicos conciencia acerca de las herramientas computacionales para clasificación automática de datos.

1.3.2. Impacto social.

Desde el punto de vista social, el agrupamiento ha tenido aplicaciones en mercadeo, donde el comportamiento de un cliente puede ser atribuido a un grupo de clientes con cierta similitud, adicionalmente esto permite generar recomendaciones (o predicciones) con las cuales se plantea atender mejor al cliente.

Otra aplicación es desde el ámbito en Ciencias Sociales, donde cada región de una determinada área puede ser definida en grupos debido a su cultura o idioma, y atribuir estás características a comportamientos específicos de ese grupo de personas.

1.4. Limitaciones

No se cuenta con unidades de procesamiento gráfico (GPU) para desarrollar el trabajo dentro un sistema embebido con funcionalidades específicas para la aplicación. Se cuenta con un equipo de costo reducido donde la capacidad de procesamiento y almacenamiento es deficiente para grandes bases de datos o procesos de mayor complejidad y costo computacional. Solo se cuentan con bases de datos libres en el repositorio (Dheeru & Karra Taniskidou, 2017) y las proporcionadas por el Grupo de Investigación y Desarrollo en Electrónica y Telecomunicaciones (GIDET).

1.5. Delimitaciones

1.5.1. Conceptual.

Se aborda con similitudes espaciales entre las muestras de datos dentro de un espacio lineal. El algoritmo está diseñado para conjuntos de datos de hasta 3 dimensiones, en el caso de los datos, hasta 3 atributos en cada dato. En el trecho de algoritmo de visión computacional se aborda solo operaciones morfológicas y de segmentación, dentro de estas se encuentran algunas de las técnicas usadas para reconocimiento de patrones y aprendizaje guiado a partir de ellos.

1.5.2. Espacial.

Es un trabajo metodológicamente limitado al espacio del globo terráqueo, pero las pruebas con datos reales están limitados al área de Colombia para bases datos en línea y el área de Cúcuta para datos adquiridos experimentalmente.

1.5.3. Temporal.

Se estima que la aprobación del proyecto tomará veinte seis (26) semanas a media jornada laboral (20 horas/semana) después de aprobado este anteproyecto.

1.6. Objetivos

1.6.1. General.

Desenvolver una metodología que permita realizar agrupamiento de datos reales con problemas de superposición (proveniente de diferentes sistemas de monitoreo o adquisición) a través de técnicas de visión computacional y métodos utilizados en el aprendizaje de máquinas.

1.6.2. Específicos.

Implementar el algoritmo de procesamiento para la corrección de la superposición, usando los softwares de Spyder y RStudio.

Experimentar con bases de datos sintéticas y reales, para grupos balanceados y desbalanceados con diferentes grados de superposición.

Seleccionar los mejores parámetros, dependiendo de la variabilidad de los datos y del valor multiplicativo en la corrección de la superposición.

Evaluar los resultados a través de métricas de precisión y exactitud, tiempo de procesamiento y costo computacional, tomando en cuenta la métrica de similitud usada y comparando con algoritmos existentes en la literatura.

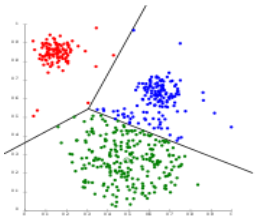
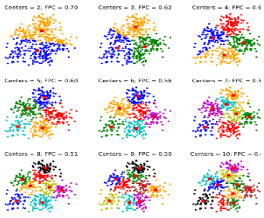
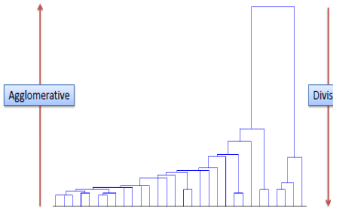
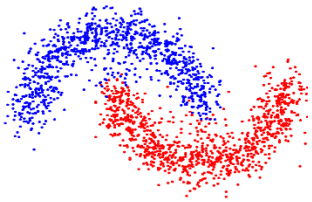
2. Marco referencial

Este capítulo presenta el estado del arte en el cual se sustentará el trabajo, tomando antecedentes de los trabajos científicos divulgados en las revistas y universidades a nivel global, las teorías establecidas y normas legales con respecto al uso de los lenguajes de programación o descripción del algoritmo.

2.1. Antecedentes

En el área de agrupamiento de datos existe un amplio conjunto de trabajo con resultados significativos para datos reales, especialmente en datos exploratorios, donde esté en un principio era calificado de una estadística inferencial avanzada debido a su fundamentación matemática.

Tabla 1.Técnicas clásicas de agrupamiento de datos.

k-means	Fuzzy cmeans	Jerárquico	Espectral
			

En la Tabla 1 se ha resumido las técnicas más populares en el agrupamiento de datos, sin embargo, estás técnicas tienen sus ventajas y desventajas dependiendo del conjunto de datos abordados. Un ejemplo claro es el k-means que presenta un algoritmo eficiente y rápido de obtener los datos pero corto para sistemas altamente variables; el fuzzy c-means es una de las técnicas más completas, a través de la lógica difusa ha logrado evitar errores por variabilidad ; la técnica de agrupamiento jerárquico suele ser altamente paramétrica, donde hay que iterar entre varios k-grupos para encontrar el punto eficiente; y finalmente, el agrupamiento espectral ha

demostrado ser de los mejores modelos de agrupamiento, pero con un alto costo computacional $O(n^3)$, donde no se lograría procesar grandes bases de datos. Por consiguiente, se presentan los siguientes trabajos relacionados específicos a abordajes en la problemática asumida.

2.1.1. Segmentation using eigenvectors: a unifying view.

En una conferencia del 1999 por Yair Weiss, se empieza a introducir el concepto del procesamiento de imagen para la segmentación en conjuntos de datos, donde a través de técnicas del principal componente de análisis, se logra un agrupamiento no supervisado con un buen desarrollo basado en auto-vectores y como la visión computacional se aborda desde la percepción de los objetos o grupos por segmentos. Los autores revisan estas técnicas bastante atractivas y el reto que conllevaba desarrollarla en el tiempo cuando se realizaron las pruebas (Weiss, 1999).

2.1.2. Identification of overlapping community structure in complex networks using fuzzy c-means clustering.

En junio de 2007 una investigación desarrollada en China por Zhang Shihua, Wang Rui-Sheng y Zhang Xiang-Sun en la que se trata un algoritmo de agrupamiento de datos utilizando métodos como función de modularidad generalizada, mapeo espectral y técnica de agrupamiento, busca diseñar un algoritmo que filtre los datos en agrupamientos según sus nodos y como estos nodos correlacionan los agrupamientos dentro del conjunto de datos, para esto se usa el concepto de agrupamiento difuso, interactuando con conjuntos de datos con superposición dentro de una red compleja, tomando en cuenta el número de instancias que presentan las muestras (Zhang, Wang, & Zhang, 2007).

2.1.3. OClustR: A new graph-based algorithm for overlapping clustering.

OClustR es un algoritmo de agrupamiento basado en gráficos para la creación agrupamientos superpuestos, introduciendo una estrategia de cobertura de gráficos y filtrado. Este algoritmo obtiene mayor precisión que los creados por algoritmos previos al trabajo verificado a través de experimentos con colecciones estándar (Pérez-Suárez, Martínez-Trinidad, Carrasco-Ochoa, & Medina-Pagola, 2013).

2.1.4. An improved overlapping k-means clustering method for medical applications.

Sina Khanmohammadi, et al., en septiembre del 2016 parten de que el agrupamiento de datos es un efectivo método para descubrir la estructura en conjuntos de datos médicos, y donde la mayoría de los algoritmos agrupamiento producen agrupamientos significativos exclusivos, en el que cada muestra puede pertenecer a un solo agrupamiento. Pero, la mayoría de los datos reales en medicina tienen información inherentemente en datos con superposición, la cual podría ser explicada mejor con métodos de agrupamientos de superposición que permiten a una muestra pertenecer a más de un grupo. Overlapping k-means (OKM) es uno de los más simples y eficientes métodos de agrupamiento con superposición, este es una extensión del tradicional algoritmo k-means, donde el método OKM también sufre de sensibilidad a los centroides iniciales de los agrupamientos. Por esto, los autores han propueto un método híbrido que combina algoritmos k-harmonic means y k-means (KHM-OKM) para superar la limitación de la sensibilidad a los centroides. La idea principal detrás del método KHM-OKM es usar la salida del método KHM para inicializar los centroides de los agrupamientos del método OKM. El método propuesto es probado utilizando la métrica FBCubed, que ha demostrado ser la medida más efectiva para evaluar los algoritmos de agrupamiento con superposición con respecto a la

homogeneidad, la integridad, la rag bag y la compensación entre el tamaño y la cantidad de agrupamientos. Según los resultados de diez conjuntos de datos médicos disponibles públicamente, el algoritmo KHM-OKM supera al algoritmo original de OKM y puede utilizarse como un método eficiente para agrupar conjuntos de datos médicos (Khanmohammadi, Adibeig, & Shanehbandy, 2017).

2.1.5. Cluster CV: Uma Abordagem de Visão Computacional para a Identificação Espacial de Agrupamentos de Dados.

En el 2017 una investigación desarrollada en la Universidad Federal de Minas Gerais (UFMG) por Brayan A. Jaimes, Cristiano L. Castro, Luis B. Torres, Gustavo L. Silva y Antonio P. Braga, en la que se envuelve un enfoque de visión computacional donde para el agrupamiento de datos con superposición sin pérdida de información. En ese trabajo usaron la distancia euclidiana y los resultados arrojaron la correcta identificación y corrección de la superposición de los datos (Jaimes, Castro, Torres, Silva, & Braga, 2017).

2.1.6. Clustervision: visual supervision of unsupervised clustering.

En el 2018, una investigación realizada por Kenney Ng, et al., construyen Clustervision, una herramienta analítica visual que ayuda a los científicos de datos a encontrar la cantidad correcta de agrupamientos para una gran cantidad de técnicas y parámetros disponibles. Adicionalmente, los usuarios pueden guiar el sistema para producir más resultados relevantes, o añadir restricciones para el agrupamiento. Este nuevo enfoque fue demostrado, usando un caso de estudio con un equipo de investigadores en el área de la medicina, y mostrar que el nuevo sistema permite a los usuarios elegir una representación efectiva de sus datos complejos (Kwon, y otros, 2018).

2.2. Marco teórico

Esta sección presenta la teoría bajo la cual se aborda la problemática del Cluster CV2, comenzando por el agrupamiento de datos, realizando énfasis en las técnicas clásicas en la literatura, seguidamente se describen las técnicas de similaridad más populares y/o usadas en la literatura, continuando el tópico de procesamiento usando visión por computadora y finalizando con el análisis de componentes principales.

2.2.1. Agrupamiento de datos (Data clustering).

La agrupación de datos es el proceso de encontrar muestras similares en particiones distintas, siendo un tipo común de aprendizaje automático no supervisado y puede ser útil para resumir datos multidimensionales complejos (Celik, 2009), llevándolos a conceptos interpretable para el análisis estadístico (Lizárraga, 2008), un ejemplo claro se ilustra en la Figura 2, donde se presenta la adquisición de datos en exámenes (Serra & Tagliaferri, 2016), a pacientes con variedad de síntomas, donde cada conjunto de exámenes construye una red de similaridad entre pacientes, para consecuentemente integrar todos los síntomas en una red (Jain, Murty, & Flynn, 1999).

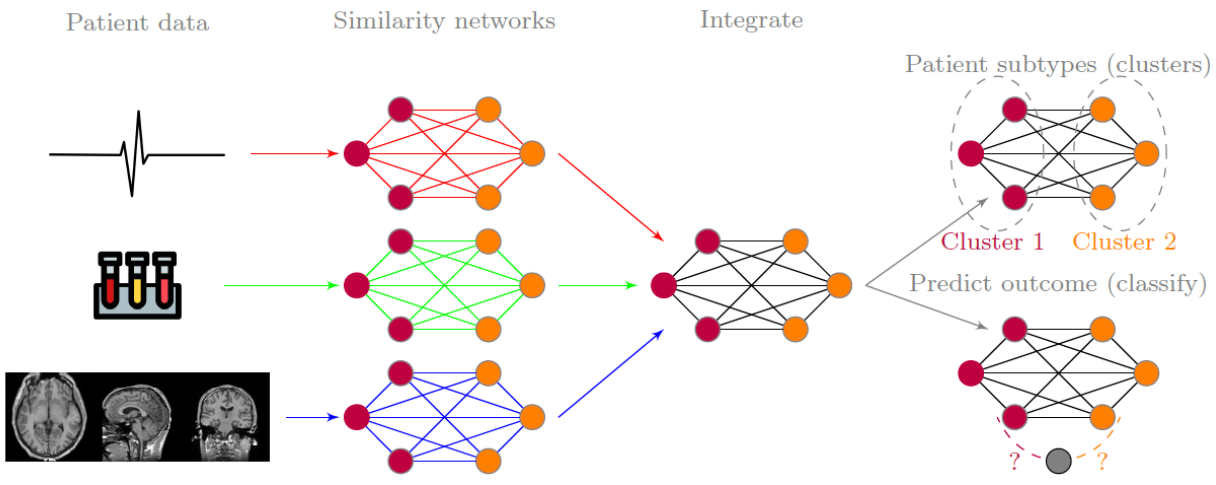


Figura 2. Ejemplo de agrupamiento de datos en Medicina.

Después de esto, se obtienen los k-grupos con los cuáles generalmente se llevan a un clasificador, donde al ingresar una nueva muestra con atribuciones similares a uno de los grupos, este ingresaría al grupo de datos (Saxena, y otros, 2017), el cual podría ser una enfermedad.

El agrupamiento de datos en términos informales consiste en encontrar los grupos ideales dentro de un conjunto de datos (Duda, Hart, & Stork, 2012), de tal manera que para un conjunto de datos definido por la ecuación:

$$D = x_1, x_2, \dots, x_n \tag{2}$$

con un número n de muestras, se debe encontrar los grupos que forman las muestras entre sí, denotados como $C_1, C_2, C_3, \dots, C_k$ para un número k de grupos (Hennig, Meila, Murtagh, & Rocci, 2015). El valor de k para una muestra x_i representa el rótulo de esta, también nombrada etiqueta, donde esta etiqueta define la afinidad de la muestra y para un conjunto de muestras con la misma etiqueta representará un grupo.

Para obtener los grupos ideales se recurre a criterios de similaridad, donde los doctores agrupan los tumores en categorías definidas por las propiedades, astrónomos agrupan las galaxias por sus formas, compañías agrupan sus clientes basado en el comportamiento o gustos, entre otros, pero bajo criterios de similaridad entre las muestras (Contreras Contreras, Medina Delgado, Guevara Ibarra, Leite de Castro, & Acevedo Jaimes, 2019).

2.2.2. Métricas de Similaridad.

Tabla 2. Métricas de distancia más usadas en la literatura.

Métrica de distancia	Modelo matemático
Euclidiana	$d_E(x_i, x_j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \cdots + (x_{in} - x_{jn})^2}$
Manhattan	$d_T(x_i, x_j) = \sum_{k=1}^{n} \lvert x_{ik} - x_{jk} \rvert$

Minkowski	$d_P(x_i, x_j) = \sum_{k=1}^{n} \left(\left\|x_{ik} - x_{jk}\right\|\right)^{1/p}$
Mahalanobis (De Maesschalck, Jouan-Rimbaud, & Massart, 2000)	$d_m(x_i, x_j) = \sum_{k=1}^{n} \frac{\left(x_{ik} - x_{jk}\right)^2}{s_k^2}$

Las métricas de similaridad permiten definir la proximidad entre muestras, dependiendo de la atribución presente en ellas y como esta atribución liga con alguno de los grupos de muestras. Para este abordaje espacial se usa métricas de distancia como método de similaridad, en la Tabla 1 se presentan las más usadas en la literatura, requeridas para este trabajo debido a su amplio estudio por parte de la distancia euclidiana, siendo el valor más representativo para la distancia espacial entre dos puntos, sin embargo la distancia manhattan maneja un método basado en el marco de referencia y depende del sistema de coordenadas cartesianas, de estás dos distancias se encuentra la Minkowski que puede recopilar las dos distancias en un solo modelo y aplicar alguna de las de las anteriores dependiendo del valor de p en el exponente de la sumatoria.

Finalmente, se cuenta con una métrica de distancia que no solo toma en cuenta la atribución del punto o espacio ocupado, sino que añade la correlación de las dos variables en cada punto a evaluar, es decir, usa la matriz de covarianza para regir el valor de similitud obtenido a través de la aplicación directa de la distancia euclidiana.

2.2.3. Visión Computacional.

La visión computacional en término informales consiste en darle a una máquina el sentido de la visión como lo tienen los humanos, para lo cual se tienen tres aspectos iniciales de la visión humana: la forma, el contexto y la cognición, donde estos pueden ser llevados a través de un

computador dentro de los términos de morfología, información a priori y patrones. Usualmente es un área pequeña dentro del procesamiento digital de señales, donde las imágenes solo son matrices donde los datos están contenidos en sus píxeles. Para este trabajo se han requerido los siguientes operadores:

2.2.3.1. Difuminación con mediana.

Sustituye por el valor de la mediana de los píxeles contenidos en la vecindad de tamaño N, al pasar por este filtro el valor final es un valor real presente en la imagen con el objetivo de reducir el efecto borroso y con poca sensibilidad para valores extremos. La mediana en estadística es definida como el dato del medio en un conjunto ordenado de datos, lo cual al ordenar los datos y determinar el punto central implica evado costo computacional cuando se tratan de imágenes que poseen gran cantidad de datos contenidos en sus píxeles.

$$\begin{bmatrix} 20 & 23 & 30 & 31 \\ 22 & 21 & 29 & 30 \\ 23 & 24 & 32 & 33 \\ 29 & 31 & 34 & 37 \end{bmatrix} \rightarrow \begin{bmatrix} - & - & - & - \\ - & 23 & 30 & - \\ - & 29 & 31 & - \\ - & - & - & - \end{bmatrix} \tag{3}$$

2.2.3.2. Binarización.

Consiste en llevar la matriz de una imagen ($I(m,n)$) a valores lógicos de unos (1) y ceros (0), donde el uno lógico representa un color blanco puro y el cero lógico representa un color negro puro. Para realizar esta transformación es necesario definir el umbral (b) de selección que determinará si un color pasa a ser uno lógico o 0 lógico dependiendo de la cantidad de componente espectral requeridos.

$$I(m,n) = \begin{cases} 0, & I(m,n) < b \\ 1, & I(m,n) \geq b \end{cases} \tag{4}$$

2.2.3.3. Operador Sobel.

Es un operador diferencial discreto que calcula una aproximación al gradiente de la función ($I(m,n)$) en función de la imagen, es decir, para cada uno de los puntos dentro de la imagen el

operador arroja el gradiente correspondiente como vector normal del punto. Con esto operador es posible determinar qué tan suave o abrupto es el cambio de intensidad en una imagen, como al mismo tiempo la dirección de cambio, para aplicar la convolución a la matriz de la imagen es necesario usar dos matrices Kernels 3×3, donde uno es para los cambios verticales (∇_y) y otro para los cambios horizontales (∇_x).

$$\nabla_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} * I \tag{5}$$

$$\nabla_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} * I \tag{6}$$

Después de obtener los gradientes en las dos direcciones se puede calcular la magnitud ∇ y dirección θ usando la siguiente ecuación.

$$\nabla = \sqrt{{\nabla_x}^2 + {\nabla_y}^2}\,, \;\; \theta = \frac{\nabla_y}{\nabla_x} \tag{7}$$

2.2.3.4. Erosión.

Es el operador morfológico que expande el valor de un pixel sobre sus alrededores, tal como se presente en la ecuación (8). En un mapa de pixeles, se expande el valor central, tomando como referencia una matriz kernel con los pesos de los valores a obtener.

$$\begin{bmatrix} - & 0 & 0 & 0 \\ - & 0 & 1 & 0 \\ - & 0 & 0 & 0 \\ - & - & - & - \end{bmatrix} \rightarrow \begin{bmatrix} - & 1 & 1 & 1 \\ - & 1 & 1 & 1 \\ - & 1 & 1 & 1 \\ - & - & - & - \end{bmatrix} \tag{8}$$

2.2.3.5. Dilatación.

Es el operador inverso a la erosión, es decir, en una región de pixeles de valores ajustados a un kernel se establecen todos los componentes y se deja el valor central como se presenta en la ecuación (9).

$$\begin{bmatrix} - & - & - & - \\ 1 & 1 & 1 & - \\ 1 & 1 & 1 & - \\ 1 & 1 & 1 & - \end{bmatrix} \rightarrow \begin{bmatrix} - & - & - & - \\ 0 & 0 & 0 & - \\ 0 & 1 & 0 & - \\ 0 & 0 & 0 & - \end{bmatrix} \quad (9)$$

En las ecuaciones 8 y 9, los valores de 0 representan el color negro y los valores de 1 representan el color blanco.

2.2.3.6. Convolución integral.

$$H = \int_{-W}^{W} G(-x) f(x) dx \quad (10)$$

2.2.3.7. Tasa de error.

$$SNR = \frac{A\left|\int_{-W}^{0} f(x)dx\right|}{n_0\sqrt{\int_{-W}^{W} f^2(x)dx}} \quad (11)$$

2.2.3.8. Localización.

$$Localization = \frac{A|f(0)|}{n_0\sqrt{\int_{-W}^{W} f^2(x)dx}} \quad (12)$$

2.2.3.9. Respuesta de bordes.

$$x_{zc} = \pi\left(\frac{\int_{-\inf}^{\inf} f(x)dx}{\int_{-\inf}^{\inf} f^2(x)dx}\right)^{\frac{1}{2}} \quad (13)$$

2.2.4. Análisis de componentes principales.

Según (Kavitha, Sandeep, & Praveen, 2016), para calcular los componentes principales sobre un conjunto de datos, se sigue el siguiente procedimiento:

Se asume un conjunto de datos descrito por sus atribuciones *[x, y, z]* y *n* muestras contenidas:

$$\mathbf{D} = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \\ y_1 & y_2 & \cdots & y_n \\ z_1 & z_2 & \cdots & z_n \end{bmatrix} \tag{14}$$

A cada una de sus atribuciones, se le obtiene la media aritmética o promedio (Navidi, 2006) usando la ecuación:

$$\begin{aligned} \mu_x &= \frac{1}{n}\sum_{i=1}^{n} x_i \\ \mu_y &= \frac{1}{n}\sum_{i=1}^{n} y_i \\ \mu_z &= \frac{1}{n}\sum_{i=1}^{n} z_i \end{aligned} \tag{15}$$

Seguidamente, cada una de las muestras, se le resta su respectiva media, con el objeto de obtener su la matriz de variabilidad respecto al centroide:

$$\mathbf{M} = \begin{bmatrix} (x_1 - \mu_x) & (x_1 - \mu_x) & \cdots & (x_1 - \mu_x) \\ (y_1 - \mu_y) & (y_1 - \mu_y) & \cdots & (y_1 - \mu_y) \\ (z_1 - \mu_z) & (z_1 - \mu_z) & \cdots & (z_1 - \mu_z) \end{bmatrix} \tag{16}$$

Luego de esto, obtener la matriz de covarianza representa obtener los productos de las variables contenidas de una variable con respecto a las otras dos.

$$\mathbf{\Sigma} = \frac{\sum_{k=1}^{m}\left(x_{ik} - \mu_{x_i}\right)\left(x_{jk} - \mu_{x_j}\right)}{m} \tag{17}$$

Este procedimiento es expresado de una manera más simple, para el caso de conjuntos de datos con más una dimensión de atribuciones, quedando como el solo producto de la matriz de variabilidad por su traspuesta:

$$\mathbf{C} = \mathbf{M}\mathbf{M}^T \tag{18}$$

Luego de esto, se plantea la suposición de que hay un solo escalar que, multiplicado por vector, resulta el mismo valor que multiplicar una matriz para ese mismo vector, es decir:

$$\mathbf{Av} = \lambda\mathbf{v} \tag{19}$$

La suposición de propuesta arriba es justo bajo la cual se logran obtener los auto valores y auto vectores de la matriz de covarianza, donde los componentes principales en este procedimiento han sido la variabilidad y dispersión de los datos con respecto al centroide.

$$\mathbf{Cv} = \lambda\mathbf{v} \tag{20}$$

Los auto vectores contienen en su diagonal la variabilidad de cada atribución, el resto son las variabilidades entre atribuciones.

2.3. Marco legal

Para este trabajo se hará uso de herramientas de software de código abierto, donde la selección es justificada en aquellos que permiten desarrollar sistemas de inteligencia artificial versátilmente y presenten toolbox considerablemente amplios para análisis de datos, visión computacional, manipulación estadística y procesamiento de señales. En este orden de ideas se proponen los lenguajes de programación Python (Python language reference, version 3.7, n.d.) y R (The R Project for Statistical Computing, 2019), estos están bajo las condiciones establecidas previamente [36-38]. En relación con esto se presenta la regulación de la licencia MIT planteada para el Cluster CV2 a nivel internacional, regulada bajo los términos de software libre (The R Project for Statistical Computing, 2019) de la Open Source Initiative (OSI) y seguidamente se presenta la normatividad colombiana de derechos de autor, donde para software libre no existe clausulas definidas de regulación, pero si definiciones acerca de lo mismo.

2.3.1. Licencia MIT.

El software se proporciona “tal cual”, sin garantía de ningún tipo, expreso o implícito incluido, pero sin limitación a las garantías de comerciabilidad, idoneidad para un propósito particular y no incumplimiento. En ningún caso, los autores o titulares de derechos de autor serán

responsables por cualquier reclamación, daños u otras responsabilidades, debido a que sea responsable de un contrato, corte u otra manera, derivados de, fuera o en conexión con el software o el uso u otras reparaciones en el software.

2.3.2. Ley 44 de 1993.

Esta penaliza el uso inadecuado de software licenciado en Colombia, donde el caso de este trabajo no es contemplado bajo la normativa debido que, a la condición de software libre, ni protege el uso de este, sin embargo, se han adoptado otras clasificaciones de semilibre, donde a pesar de la condición de libre uso este no desmerita el autor de su obra, tal como lo hace la licencia MIT.

3. Metodología

La metodología propuesta para el desarrollo del Cluster CV2 se fundamenta en la ingeniería de software con el paradigma de programación orientado a objetos (POO) y en metodologías ágiles para el desarrollo de software (Mantilla, Ariza, & Delgado, 2014). En la Figura 3 se presenta la metodología enmarcada en cinco fases denominadas: análisis, diseño, desarrollo, pruebas de funcionamiento y entrega, donde para cada una de las fases, se realiza una descripción detallada de las actividades que intervienen en el desarrollo y las condiciones o parámetros iniciales que se deben tener en cuenta en cada una de ellas.

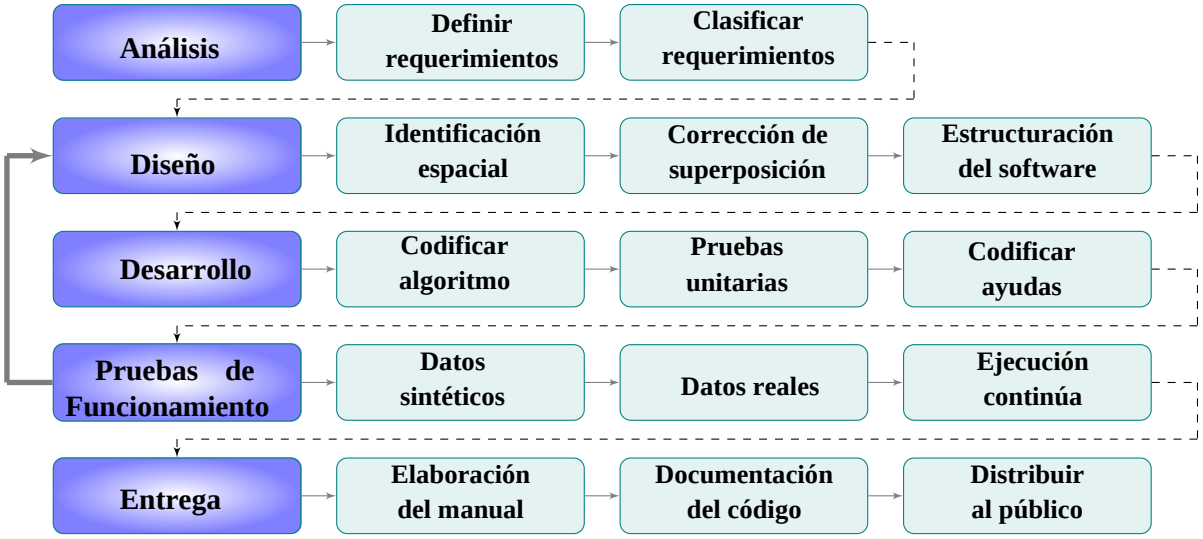


Figura 3. Etapas de la metodología para el desarrollo del Cluster CV2.

3.1. Análisis de requerimientos

El éxito de un proyecto de implementación de software depende de un buen proceso de levantamiento de requerimientos y un buen entendimiento del producto, para lograrlo, todos los interesados en el proyecto deben ver reflejadas sus necesidades e intenciones en el producto terminado, para ello se requiere la utilización de modelos entendibles y estandarizados. Así, los requerimientos del proyecto han sido definidos al igual que su impacto en la solución de las problemáticas planteadas y las cuales consisten en:

- Identificar los k-grupos contenidos en un conjunto de datos sin información a priori.
- Corregir la superposición entre grupos de datos.

Para analizar los requerimientos, se han realizado dos pasos: la definición y la clasificación, esto pasos permiten estructurar los pilares de la propuesta y cuál es la función de cada uno de ellos dentro de la misma.

3.1.1. Definición.

En el desarrollo de software embebido se encuentra las características confiabilidad $R(t)$, mantenibilidad $M(d)$, disponibilidad $A(t)$, seguridad y eficiencia, dichas permiten valorar el comportamiento del software. Por lo tanto, en función las características del software embebido se han propuesto los siguientes requerimientos:

Cluster CV2 debe identificar espacialmente los grupos contenidos en un conjunto de datos usando visión computacional sobre una matriz de similaridad.

La matriz de similaridad debe representar la relación entre atribuciones usando valores entre 0 a 1.

El único parámetro permisible para la identificación espacial, debe ser un límite de variabilidad (L), el cual puede ser obviado al 1.3 cuando el usuario no lo indique

El único parámetro permisible para la corrección de la superposición es el factor multiplicativo (q), con el cual se trasladen los grupos de posición.

Cluster CV2 debe almacenar la matriz de similaridad en formato PNG, con el objeto de visualizar la similaridad y generar un k intuitivo a obtener cuando alguien se encuentre supervisando la identificación.

Cluster CV2 debe ser desarrollado bajo el paradigma POO, el cual permite reducir el tiempo de mantenimiento y evidencia claramente su estructura.

Los datos sin superposición deben ser proyectados dentro de un nuevo espacio lineal, evitando la pérdida de información y salvando dichos datos en un archivo .CSV.

Todo el trabajo debe ser desarrollado bajo código abierto, permitiendo al público realizar cambios para sus aplicaciones específicas.

Cluster CV2 debe contener ejemplos o rutinas de implementación sobre estructuras de datos, las que permitan evidenciar claramente el uso del software.

Cluster CV2 debe presentar una documentación, donde se describan los métodos, entradas y salidas, que conlleva cada una de sus instrucciones.

3.1.2. Clasificación.

Por consiguiente, los requerimientos generan un impacto en el desarrollo del proyecto con el cual se determinará el tiempo de ejecución necesario para cumplir con el mismo, asumiendo una ruta crítica de aquellos que son indispensables. Sin embargo, esto no implica que los demás requerimientos carezcan de relevancia, es así como se ha determinado en la Tabla 3 cuales son de aspecto técnico, cuales son de aspecto funcional y cuales son de aspecto no funcional (Molina & Moreno, 2010). Los requerimientos funcionales son todos aquellos que demandan una función dentro del sistema, los no funcionales son aquellos que no interfieren con el alcance de las metas y los técnicos son aquellos que permite al Cluster CV2 ser escalable en versiones de recursos y dispositivos de uso (Chaves, 2005).

Tabla 3. Requerimientos del Cluster CV2.

Descripción	Técnico	Funcional	No funcional
Identificar espacialmente *k*-grupos, usando visión computacional	xx	xx	
La matriz de similaridad debe presentar la relación normalizada entre las muestras		xx	
Los parámetros opcionales de entrada son el limiar y el factor multiplicativo	xx		
La matriz de similaridad debe ser salvada en archivo PNG			xx
Los datos sin superposición deben ser proyectados en un nuevo espacio lineal			xx
Todo el trabajo debe estar desarrollado bajo código libre	xx		
Añadir ejemplos para datos sintéticos y reales			xx
Documentar todos los métodos diseñados	xx		xx

3.2. Diseño

Teniendo claro los requerimientos del ClusterCV, para su diseño se han abordado tres etapas: la identificación espacial, en esta se describen el procedimiento matemático propuesto para la identificación usando visión computacional, donde un conjunto de datos al ser proyectados usando la matriz de afinidad, es convertida a una imagen con la información relevante acerca de las similaridades presentes en el conjunto de datos; la corrección de la superposición, para esta se plantea un enfoque de descomposición de la matriz de covarianza, en donde se logra determinar la dirección y magnitud de propagación de un grupo de datos, con el objeto de reconstruir el conjunto de datos en un nuevo espacio lineal sin superposición; finalmente se tiene la estructuración del software, en esta sección se describe la conexión de las dos funciones principales y como el software ha sido estructurado, resaltando los criterios bajo los cuales se ha definido.

3.2.1. Identificación de los agrupamientos por visión computacional.

Sea **D** un conjunto de datos obtenidos en una investigación hipotética, para la cual se ha definido como:

$$\mathbf{D} = \{\mathbf{x_1}, \mathbf{x_2}, \mathbf{x_3}, \mathbf{x_4}, \cdots, \mathbf{x_m}\} \tag{21}$$

donde cada $\mathbf{x_i}$ es un dato correspondiente a **D** con *m* dimensiones, es decir, $\mathbf{x_i} \in \Re^m$. Para este tipo de estructura de datos, donde sus atribuciones son netamente de tipo numérico real, se ha planteado la identificación de los grupos usando los procedimientos de la Figura 4, donde el conjunto de datos pasa por 4 procedimientos descritos a detalle en las siguientes secciones, tomando en cuenta de que cada procedimiento debe ser realizado justo después de que el previo ha sido completado.

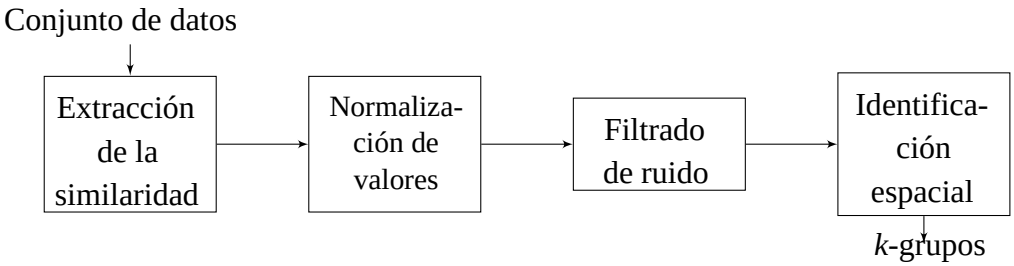


Figura 4. Identificación de los k- grupos usando la matriz de similaridad.

3.2.1.1. Matriz de similaridad.

Tomando el conjunto de datos **D** como una matriz de $m \times n$ términos, siendo m el número de filas (número de datos) y n el número de columnas (número de atribuciones), existe una función $f(i,j)$ que representa la similitud entre muestras, de manera que con esta se obtiene una matriz $\mathbf{M}(\mathbf{i},\mathbf{j})$ expresada en la ecuación (22).

$$\mathbf{M}(\mathbf{i},\mathbf{j}) = \begin{bmatrix} f(1,1) & f(1,2) & \cdots & f(1,n) \\ f(2,1) & f(2,2) & \cdots & f(2,n) \\ \vdots & \vdots & \ddots & \vdots \\ f(m,1) & f(m,2) & \cdots & f(m,n) \end{bmatrix} \tag{22}$$

Esta función ($f(i,j)$) generalmente es una métrica de distancia como las que fueron mencionadas en la Tabla 4, bajo la condición de que, para convertirse en función, estás deben ser aplicadas par-a-par entre las muestras, ¿cómo es esto? Para cada muestra contenida en el conjunto de datos se aplica la métrica de distancia sobre todas las muestras, tal como se haría en el agrupamiento espectral cuando se usa la métrica de la ecuación (23) aplicable para valores $i \neq j$ y $A_{ii} = 0$ en una matriz de afinidad $A \in \Re^{\mathrm{m}}$ (Kavitha, Sandeep, & Praveen, 2016).

$$A_{i,j} = \exp\left(\frac{||s_i - s_j||^2}{2\sigma^2}\right) \tag{23}$$

Para este trabajo, hemos establecido el tipo de identificación y las métricas usadas en la Tabla 4, partiendo de las conclusiones obtenidas en los trabajos de (Perlibakas, 2004), estos

manejan muchos más enfoques de distancia o modificados con el objeto de encontrar elevar la precisión o validez en sus resultados.

Tabla 4. Tipo de identificaciones obtenidas por las métricas de distancia.

Métrica de distancia	Modelo matemático	Tipo de identificación
Euclidiana	$d_E(x_i, x_j) = \sqrt{\sum_{k=1}^{n} (x_{ik} - x_{jk})^2}$	Enfoque espacial
Manhattan	$d_T(x_i, x_j) = \sum_{k=1}^{n} \lvert x_{ik} - x_{jk} \rvert$	Por linealidad
Minkowski	$d_P(x_i, x_j) = \sum_{k=1}^{n} \left(\lvert x_{ik} - x_{jk} \rvert^p \right)^{1/p}$	Valores multidimensionales
Mahalanobis	$d_m(x_i, x_j) = \sum_{k=1}^{n} \frac{\left(x_{ik} - x_{jk}\right)^2}{s_k^2}$	Media y varianza

Teniendo definido el concepto de matriz de afinidad y como esta actúa con la métrica de distancia seleccionada, la matriz de la ecuación (22) quedaría de la forma de la ecuación (24), siendo sus diagonales 0 por ser la misma muestra, donde para cada muestra debe existir la relación de ella con respecto a todas las demás muestras contenidas en el conjunto de datos. Adicionalmente se resalta que la mayor similaridad se obtiene con muestras del mismo valor (Weinberger & Saul, 2009), es decir, el caso $i = j, s_i = s_j \Rightarrow s_i - s_j = 0$.

$$\mathbf{M} = \begin{bmatrix} 0 & \sqrt{\sum_{k=1}^{K}(x_{1k} - x_{2k})^2} & \cdots & \sqrt{\sum_{k=1}^{K}(x_{1k} - x_{nk})^2} \\ \sqrt{\sum_{k=1}^{K}(x_{2k} - x_{1k})^2} & 0 & \cdots & \sqrt{\sum_{k=1}^{K}(x_{2k} - x_{nk})^2} \\ \vdots & \vdots & \ddots & \vdots \\ \sqrt{\sum_{k=1}^{K}(x_{mk} - x_{1k})^2} & \sqrt{\sum_{k=1}^{K}(x_{mk} - x_{2k})^2} & \cdots & 0 \end{bmatrix} \tag{24}$$

3.2.1.2. Normalizado.

Los valores obtenidos en la matriz **M** varían dependiendo de las magnitudes usadas en el conjunto de datos, para lo cual se ha requerido el normalizado de los datos dentro de una escala estándar con el objeto de abordar cualquier tipo de magnitud. En un principio, este procedimiento está orientado a evidenciar un pequeño cambio en los valores de la matriz **M**, tal y como un diferencial parcial lo describe en la ecuación (25), donde el primer término entre paréntesis indica el valor de normalizado y el segundo término a la derecha, va presentando los pequeños cambios.

$$\left(\frac{\text{MEAN}\{\mathbf{M}\}}{\text{MAX}\{\mathbf{M}\} - \text{MIN}\{\mathbf{M}\}}\right)\left(\frac{\partial \mathbf{M}(\mathbf{i}, \mathbf{j})}{\partial \mathbf{x_{ij}}}\right) \tag{25}$$

En el caso de los conjuntos de datos, las variaciones suelen ser representadas por diferencias, debido a que otro tipo de variaciones asume criterios más críticos e innecesarios para el normalizado. Por esto, la normalización ha sido llevada al modelo descrito en la ecuación (26), donde la parte derecha de la igualdad es la analogía con respecto a la ecuación (25), y en donde se pueden identificar el término de normalizado y la pequeña variación.

$$\mathbf{N}(\mathbf{i},\mathbf{j}) = \left(\frac{1}{M_{max} - M_{max}}\right)(\mathbf{M} - M_{min}) \tag{26}$$

Aunque los valores se encuentren normalizados para la ecuación (26), la escala permanece incierta, debido a que esta depende de los valores máximos y mínimos presentes en **M**. Por lo tanto, la escala ha sido regulada al añadir un valor mínimo de muestra S_{min} y un valor máximo de muestra S_{max}, estos permiten ajustar la escala con solo añadirlos a la ecuación (26), lo cual ha resultado en la ecuación (27).

$$\mathbf{MN}(\mathbf{i},\mathbf{j}) = S_{min} + S_{max}\mathbf{N} = S_{min} + S_{max}\left(\frac{\mathbf{M} - M_{min}}{M_{max} - M_{max}}\right) \tag{27}$$

En el caso particular del Cluster CV2, los valores de S_{min} y S_{max} son asumidos como 0 y 1, respectivamente. Este último procedimiento aparte de normalizar los valores en las muestras, también nos ha permitido establecer el rango al cual la matriz $\mathbf{x_{ij}}$ se ha convertido.

3.2.1.3. Filtrado de ruido.

La matriz **NM** presenta la información visual obtenida directamente del conjunto de datos con los valores estandarizados, sin embargo, en esta se encontró la presencia de ruido, por decir ruido a aquellos valores cuyas afinidades no representan relevancia para la identificación de los k-grupos contenidos dentro del conjunto de datos. El ruido es sencillamente zonas donde la afinidad se refleja, pero no es certero para el agrupamiento de los datos, con el objeto de reducir el ruido se ha desarrollado dos procedimientos, uno estadístico y otro por visión computacional. Para el caso del procedimiento estadístico, se ha recurrido al concepto de desviación estándar, esta consiste en describir el grado de dispersión bajo el cual se encuentra sometido un dato, es decir, que tanto varía el mismo de la media (Hernández Sampieri, Fernández Collado, & Baptista Lucio, 2003). La desviación estándar es calculada usando la ecuación (28), donde μ es la media

aritmética de los datos, m es el número de datos y cada ith término es cada uno de los datos al iterar la sumatoria.

$$s_d = \sqrt{\frac{\sum_{k=1}^{m}(\mathbf{x} - \mu_x)}{m}} \tag{28}$$

Esta medida da una certeza de que tanto varían las muestras con respecto al punto central, en este caso la media, con la cual se ha diseñado la función de filtrado de variabilidad, esta es descrita en la ecuación (29), esta es función de la matriz normalizada y un valor límite L, actuando como multiplicativo a la desviación estándar del conjunto de datos.

$$\mathbf{L}(\mathbf{x_i}, \mathbf{x_j}) = \begin{cases} 1, & \text{si} \quad \mathbf{NM}(\mathbf{x_i}, \mathbf{x_j}) \geq L * s_d \\ \mathbf{NM}(\mathbf{x_i}, \mathbf{x_j}), & \text{si} \quad \mathbf{NM}(\mathbf{x_i}, \mathbf{x_j}) < L * s_d \end{cases} \tag{29}$$

A través de esta función se intenta reducir la complejidad en la detección de los grupos, aunque L es un valor paramétrico desconocido, pruebas preliminares arrojaron reducción del ruido en $L = [0.9,1.7]$ para distribuciones normales gaussianas, estos valores han evidenciado de 35% al 68% de variabilidad, lo cual ha permitido concluir en dejar $L = 1.3$ para valor por defecto. Por otro lado, se ha desarrollado una reducción de ruido usando operadores matriciales de visión computacional, en este caso, se ha requerido llevar la matriz $\mathbf{L}(\mathbf{x_i}, \mathbf{x_j})$ a una imagen, de manera que se opere a través de ventanas (kernels) y usando la difuminación mediana sobre la imagen, descrita en la ecuación (30).

$$\mathbf{G}(\mathbf{x_i}, \mathbf{x_j}) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x_i^2 + x_j^2}{2\sigma^2}\right) \tag{30}$$

Los i y j enésimos términos representan la coordenada (i, j) dentro de la matriz $\mathbf{L}$, dando para cada uno un valor de ruido $\sigma = 5$, inicialmente abordado sobre ese valor para el cual

equivale a un kernel de 20×20 píxeles, sobre el peso de la ventana $p^2 = 400$ descritos en la ecuación (31).

$$K = \frac{1}{400}\begin{bmatrix} 1_{1\times1} & 1_{1\times2} & \cdots & 1_{1\times20} \\ 1_{2\times1} & 1_{2\times2} & \cdots & 1_{2\times20} \\ \vdots & \vdots & \ddots & \vdots \\ 1_{20\times1} & 1_{20\times2} & \cdots & 1_{20\times20} \end{bmatrix} \tag{31}$$

Este valor ha sido determinado experimentalmente al realizar 10 pruebas con 2 distribuciones normales gaussianas desbalanceadas, las pruebas no definieron el valor exacto del kernel, sino el porcentaje de esta ventana con respecto a la cantidad de muestras contenidas en el conjunto de datos, la ecuación (31) representa los resultados con 900 muestras. Repitiendo esta prueba con 5 distribuciones normales gaussianas, se encontró un ruido homogenizado similar a la prueba con 2 de estas. Bajo estos dos parámetros, se ha definido que la ventana para homogenizar el ruido sobre la imagen debe ser el valor más cercano al 2% de la cantidad del conjunto de datos y dando como caso general de kernel la ecuación (32),

$$h = \text{int}\{0.02 * m\}, \qquad K(m) = \frac{1}{h^2}\begin{bmatrix} 1_{1\times1} & 1_{1\times2} & \cdots & 1_{1\times h} \\ 1_{2\times1} & 1_{2\times2} & \cdots & 1_{2\times h} \\ \vdots & \vdots & \ddots & \vdots \\ 1_{h\times1} & 1_{h\times2} & \cdots & 1_{h\times h} \end{bmatrix} \tag{32}$$

donde int{} es una función que retorna el valor entero del valor que se encuentre contenido en las llaves, h es el parámetro bajo el cual se dimensiona la ventana del kernel, adicionalmente a esto, el kernel planteado se ha asumido en 1 para cada valor de la matriz en la coordenada (i, j) y el cual corresponde a su peso total h^2.

3.2.1.4. Reconocimiento por visión computacional.

En el filtrado de ruido se ha evidenciado las facilidades de implementar operadores digitales sobre señales como lo es un filtro, sin entrar mucho en detalle acerca de la respuesta obtenida, sino que enfocado a lograr la tarea para la cual se ha diseñado, por consiguiente fue uno de los

motivos por los cuales fue propuesto el enfoque de visión computacional, en este caso, la Figura 5 presenta un modelo básico de imagen digital, donde un punto en la imagen representa la función de una par de dos valores (coordenada) y esta a su vez es una matriz de valores donde sus valores representan un color, tanto como la posición en la matriz indica la coordenada en el plano.

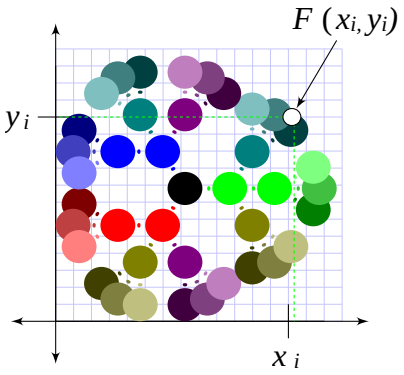


Figura 5. Representación de una imagen digital como función F(x,y).

El concepto de coordenada es una pareja de números que se ha usado para determinar el punto de encuentro entre dos conjuntos de valores (conjunto de datos), es decir, la función $F(x_i, y_i)$ representa la relación existente entre la muestra x_i y y_i, o más bien en términos del Cluster CV2, la función es la métrica de distancia y los valores de la coordenada es un dato con respecto a otro conjunto del mismo dato. A través de esa función determinada generalmente por el color presente en la imagen, es que se logra determinar la presencia de agrupamientos, sin embargo, abordar un esquema de colores de tres canales como el presentado en la Figura 5, requiere de mayor costo computacional cuando solo se trata de una característica como la similaridad entre muestras.

$$\boldsymbol{I_b(i,j)} = \begin{cases} \mathbf{0}, & \boldsymbol{G * L < b} \\ \mathbf{1}, & \boldsymbol{G * L \geq b} \end{cases} \tag{33}$$

Para reducir a un solo canal se ha optado por la binarización (presentada en la ecuación (33)), esta tiene un comportamiento similar al expresado en la función de filtrado de ruido usando la desviación estándar (véase la ecuación 29), aunque para este caso el criterio es el color presente en cada enésimo término de la matriz. La ecuación (33) presenta un valor b, donde si bien este tiene una analogía con el valor L de la ecuación (29), la tarea de ambos está conectada, puesto que los dos son límites pero uno es respectivo al análisis estadístico y el otro es respecto al análisis de visión computacional, donde este último también ha sido definido experimentalmente, debido a que las zonas donde $b = 0.75$ del pigmento del color negro o mayor en este valor, esta será la zona que debe conservarse como 1, y para otros casos se lleva a 0. Luego de esto, solo quedan pequeñas zonas de la imagen donde el ruido permanece, dichos valores deben ser tan pequeños que bastaría con aplicar el operador morfológico *opening*, el cual es definido en la ecuación (34).

$$A \circ B = (A \ominus B) \oplus B \tag{34}$$

El operador morfológico opening consiste en realizar la erosión justo después de la dilatación, asumiendo el mismo kernel propuesto por la ecuación (32) y permitiendo eliminar las pequeñas inserciones de ruido (Parker, 2010). Finalmente, para la detección de los grupos basta con conocer el número de objetos en la imagen, puesto que para este punto solo las zonas con alto grado de similaridad han de permanecer en la misma, en esta tarea se ha abordado el algoritmo canny, el cual usa solo cuatro pasos para la detección de los bordes presentes en la imagen (Tan, Isa, & Lim, 2013):

Remover el ruido presente usando la máscara de la ecuación (32).

Encontrar los gradientes diferenciales ∇_x y ∇_y orientados en las direcciones de máximo cambio para x y y, respectivamente.

Supresión no máxima, implicando la interpolación en subpíxeles de las intensidades de píxel.

Umbralización de la histéresis.

En este punto, el detector canny ha arrojado las ubicaciones de los objetos o bien definidos k-grupos contenidos en el conjunto de datos, donde no se le ha suministrado parámetros acerca de cuantos grupos o información que limite el número de grupos a encontrar en el conjunto de datos.

3.2.2. Corrección de la superposición.

Con los grupos identificados, queda la problemática de superposición entre estos grupos, para esto se ha propuesto usar la descomposición de la matriz de covarianza en auto valores y auto vectores, donde la matriz de covarianza contiene la información relacionada a la dispersión de los datos, seguidamente esta se descompone en auto valores y auto vectores (eigenvalues, eigenvectors) usando el análisis de componentes principales, los auto vectores dan información acerca de las direcciones de propagación y dispersión, con los cuales se modifican los valores dentro del conjunto de datos, tomando un grupo a la vez, trasladándolo y repeliéndolo de los demás grupos, bajo parámetros de factores correctivos, definidos experimentalmente.

3.2.2.1. Matriz de covarianza.

Para aquellos datos con n dimensiones, siendo la ecuación (35) una matriz de diferencias entre el conjunto de datos $\mathbf{D}(\mathbf{x_i}, \mathbf{x_j}, \cdots, \mathbf{x_n})$ con m-número de datos (descrito previamente en la ecuación (21)) y sus medias aritméticas $\mu_{x_i}, \mu_{x_j}, \cdots, \mu_{x_m}$.

$$\mathbf{M_\Delta} = \begin{bmatrix} (x_{11} - \mu_{x_1}) & (x_{12} - \mu_{x_1}) & \cdots & (x_{1n} - \mu_{x_1}) \\ (x_{21} - \mu_{x_2}) & (x_{22} - \mu_{x_2}) & \cdots & (x_{2n} - \mu_{x_2}) \\ \vdots & \vdots & \ddots & \vdots \\ (x_{m1} - \mu_{x_m}) & (x_{m2} - \mu_{x_m}) & \cdots & (x_{mn} - \mu_{x_m}) \end{bmatrix} \tag{35}$$

Estas diferencias representan la dispersión, donde la varianza implica elevar cada uno de los ith y jth términos de la matriz, para luego obtener la sumatoria. Sin embargo, cuando se desconoce el número de dimensiones de los datos (determina por n), este análisis es impreciso en definir la dispersión de los datos, la matriz de covarianza $\mathbf{\Sigma}$ se obtiene multiplicando la matriz de diferencias por su traspuesta, tal como se presenta en la ecuación (36).

$$\mathbf{M_\Delta} \cdot \mathbf{M_\Delta}^T = \begin{bmatrix} (x_{11}-\mu_{x_1})^2 + (x_{12}-\mu_{x_1})(x_{21}-\mu_{x_2}) + \cdots + (x_{1n}-\mu_{x_1})(x_{m1}-\mu_{x_m}) & (x_{11}-\mu_{x_1})(x_{12}-\mu_{x_1}) + (x_{12}-\mu_{x_1})(x_{22}-\mu_{x_2}) + \cdots + (x_{1n}-\mu_{x_1})(x_{m2}-\mu_{x_m}) & \cdots & (x_{11}-\mu_{x_1})(x_{1n}-\mu_{x_1}) + (x_{12}-\mu_{x_1})(x_{2n}-\mu_{x_2}) + \cdots + (x_{1n}-\mu_{x_1})(x_{mn}-\mu_{x_m}) \\ (x_{21}-\mu_{x_2})(x_{11}-\mu_{x_1}) + (x_{22}-\mu_{x_2})(x_{21}-\mu_{x_2}) + \cdots + (x_{2n}-\mu_{x_2})(x_{m1}-\mu_{x_m}) & (x_{11}-\mu_{x_1})(x_{12}-\mu_{x_1}) + (x_{12}-\mu_{x_1})(x_{22}-\mu_{x_2}) + \cdots + (x_{1n}-\mu_{x_1})(x_{m2}-\mu_{x_m}) & \cdots & (x_{11}-\mu_{x_1})(x_{1n}-\mu_{x_1}) + (x_{12}-\mu_{x_1})(x_{2n}-\mu_{x_2}) + \cdots + (x_{1n}-\mu_{x_1})(x_{mn}-\mu_{x_m}) \\ \vdots & \vdots & \ddots & \vdots \\ (x_{m1}-\mu_{x_m})(x_{11}-\mu_{x_1}) + (x_{m2}-\mu_{x_m})(x_{21}-\mu_{x_2}) + \cdots + (x_{mn}-\mu_{x_m})(x_{m1}-\mu_{x_m}) & (x_{m1}-\mu_{x_m})(x_{12}-\mu_{x_1}) + (x_{m2}-\mu_{x_m})(x_{22}-\mu_{x_2}) + \cdots + (x_{mn}-\mu_{x_m})(x_{m2}-\mu_{x_m}) & \cdots & (x_{m1}-\mu_{x_m})(x_{1n}-\mu_{x_1}) + (x_{m2}-\mu_{x_m})(x_{2n}-\mu_{x_2}) + \cdots + (x_{mn}-\mu_{x_m})^2 \end{bmatrix} \tag{36}$$

3.2.2.2. Descomposición en auto valores y auto vectores.

Continuando, se descompone la matriz de covarianza $\mathbf{\Sigma}$ en sus auto-valores y auto-vectores, con el procedimiento indicado en la sección 2.2.4 y llevado en contexto a la ecuación (37), siendo $\mathbf{\Sigma}$ la matriz dirección al auto-vector $\mathbf{v}$ y λ la constante (auto-valor) que modifica las direcciones de $\mathbf{v}$, tal como lo realiza la multiplicación por la matriz $\mathbf{\Sigma}$.

$$\mathbf{\Sigma} = \mathbf{M_\Delta} \cdot \mathbf{M_\Delta}^T, \qquad \mathbf{\Sigma v} = \lambda \mathbf{v} \tag{37}$$

A este punto, en $\mathbf{v}$ se encuentra un vector con dimensiones $n \times n$, el cual contiene n componentes principales de análisis expresados cada uno en función de los otros componentes. Así, λ es el conjunto de auto valores para cada auto vector de cada componente que modifica la dirección del vector $\mathbf{v}$ como lo hace la multiplicación con la matriz $\mathbf{\Sigma}$.

3.2.2.3. Traslación del enésimo grupo k.

Por lo tanto, en un análisis inicial se ha partido del hecho de corregir la superposición a datos bidimencionales, con los cuales solo bastaría con obtener dos ángulos resultantes de dos componentes principales como se presenta en la ecuación (38).

$$\theta_{PC1} = \arctan\left(\frac{v_{11}}{v_{21}}\right), \qquad \theta_{PC2} = \arctan\left(\frac{v_{12}}{v_{22}}\right) \tag{38}$$

Ahora, basta conocer el centroide general del conjunto de datos y el centroide específico para un grupo, siendo este grupo de datos el conjunto definido previamente como **D**, el cual ha sido renombrado como subconjunto de datos $\mathbf{D_1}$ y pertenece a un conjunto de datos **DS**, de manera que este conjunto de datos está definido por $\mathbf{DS} = \{\mathbf{D_1}, \mathbf{D_2}, \cdots, \mathbf{D_k}\}$, siendo k el número de grupos contenidos por **DS** (subconjuntos).

En la ecuación (39) se presenta la matriz de escala dependiente de los centroides y características de cada grupo, debido a que esta matriz recorre las muestras de cada grupo en un factor específico, es decir, la magnitud de los corrimientos aplicados.

$$\mathbf{S} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \tag{39}$$

La matriz de escala por sí sola no logra realizar un corrimiento idóneo, debido a que solo es la magnitud de este, por lo cual en la ecuación (40) se presenta la matriz de rotación, la cual usa los ángulos de los auto vectores con el objeto de conocer de desplazar el grupo de datos justo en la dirección en la cual ellos se están propagando y dispersando.

$$\mathbf{R} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \tag{40}$$

Finalmente, se obtiene la matriz de traslación (en la ecuación (41)), la cual es aplicada sobre el grupo (subconjunto) de datos con el objeto de reducir la distancia entre muestras de un mismo grupo.

$$\mathbf{T} = \mathbf{RS} \tag{41}$$

3.2.2.4. Repulsión del enésimo grupo k.

Continuando, los valores de la traslación salvados en un nuevo grupo de datos pueden haber modificado su punto central, para esto, se vuelve a calcular el centroide, tomando el centroide general del conjunto de datos. En primera instancia se propone un valor multiplicativo q, el cual es añadido en base al centroide general C_T y el centroide del grupo C_j, con la distancia de estos centroides, se eleva q veces esa distancia y de esa manera alejar el grupo de datos de la tendencia central del conjunto como es presentado en la ecuación (42).

$$d(C_i) = \sqrt{\left(C_T - C_j\right)^2} * q, q > 1 \tag{42}$$

Sin embargo, para validar este método se propone el puntaje Fisher (Tsuda, Kawanabe, & Müller, 2003) para evaluar el nivel de superposición justo después de los procedimientos de corrección, debido a que esta métrica (o kernel) ha incrementado como un extractor de características para agrupamientos, esta deriva del modelo probabilista **loglikelihood** para un parámetro de estimación $\widehat{\boldsymbol{\theta}}$ de las muestras y presentando su expresión matemática en la ecuación (43):

$$\mathbf{f}(x) = \left[\frac{\partial \log q\left(x|\widehat{\boldsymbol{\theta}}\right)}{\partial \theta_1}, \cdots, \frac{\partial \log q\left(x|\widehat{\boldsymbol{\theta}}\right)}{\partial \theta_M}\right]^T \tag{43}$$

Con el objeto de agrupar satisfactoriamente, con el puntaje Fisher se está aplicando el mapeo en un nuevo espacio lineal, bajo la limitante de pocas dimensiones para las muestras contenidas en el conjunto de datos.

3.2.3. Estructuración del software.

Sin embargo, lo anteriormente mencionado, es el diseño preliminar puesto que en el gráfico 3.1 se presentan iteraciones en el momento que se van realizando las pruebas sobre el algoritmo con respecto a costo computacional y eficiencia en la identificación usando las métricas de validación. Continuado con la ejecución, se ha diseñado los pseudocódigos para cada parte del Cluster CV2, tratando como independiente la identificación de la corrección de la superposición entre muestras, pues si bien el primero está ampliamente abordado en la literatura (Galluccio, Michel, & Comon, 2005), no siempre se logran los resultados esperados. Por otro lado, el proceso de corrección de la superposición es opcional, pues no todas las bases de datos requieren de identificación en grupos separados, sino que se deja con el objeto de llegar a aquellas aplicaciones de donde surgió la problemática.

3.2.3.1. Obtención de la matriz de similaridad.

Debido a que se han planteado dos lenguajes de programación para el desarrollo del código, se han definido los algoritmos para las acciones que cumplen los dos, sin importar la función que se use en la programación y enfocando al cumplimiento de la tarea asignada. En primera instancia se realiza la lectura del conjunto de datos, seleccionando las atribuciones cuantitativas a usar en el agrupamiento, como se expresa en el Figura 6, asumiendo que el conjunto de datos fue previamente guardado en la máquina de ejecución usando el formato de valores separados por coma (Comma Separated Values - CSV).

Algoritmo 1 Preparación del conjunto de datos

```
1: procedure PREPARINGDATASET(filename, atributions)
2:     data ← read.file(file_name)          ▷ Lee la información del archivo
3:     data ← data[atributions]
4: end procedure
```

Figura 6. Algoritmo de preprocesamiento del conjunto de datos.

Estos datos generados, deben ser valores cuantitativos de las muestras y organizadas cada una de las atribuciones por columna, donde cada muestra es representada por una fila y sus las dimensiones del conjunto de datos queda representado en las dimensiones de la matriz $m \times n$. Seguidamente, de haber aclarado los parametros de ingreso al Cluster CV2, se ha diseñado los procedimientos para obtener la matriz de distancias usando las métricas euclidianas (ver Algoritmo2), manhattan (ver Algoritmo 3), minkowsi (ver Algoritmo 4) y mahalanobis (ver Algoritmo 5). La distancia euclidiana es calculada usando el cuadrado de las diferencias de cada una de las atribuciones entre dos muestras, para el caso del Algoritmo 2 se ha definido una muestra de prueba que itera a través del resto de muestras y con cada una de sus atribuciones se obtiene la diferencia, salvando esta diferencia en una matriz nombrada $matrixEuclidean$, la cual para un cojunto de datos $m \times n$ resultará en una matriz de la forma $m \times m$, siendo el número de muestras (número de filas) y n el número de atribuciones, en base a lo previsto de las Tablas 2 y 4.

Para el Algoritmo 3 se ha recuperado la metodología usada en la distancia euclidiana, con la conveniencia que para este caso se ha usado la diferencia absoluta sin potenciar los valores sumando cada uno de ellos con el objeto de identificar que tan separados se encuentran respecto a su eje (de la atribución específica) y de esta manera obtener un valor ajustado las escalas usadas en los datos.

Algoritmo 2 Matriz de distancias euclidiana

```
1: m, n ← x.shape()
2: matrixEuclidean ← [empty − array, shape(m, n)]
3: for i ← 0, i++, i < m do                          ▷ Iterando sobre cada muestra de prueba
4:     for j ← 0, j++, j < m do                      ▷ Iterando sobre el resto de muestras
5:         suma, diferencia, potencia, distancia ← 0  ▷ (Re)inicia las variables
6:         for k ← 0, k++, k < n do                  ▷ Iterando sobre cada atributo
7:             diferencia ← x[i, k] − x[j, k]        ▷ Calcula la diferencia
8:             potencia ← (diferencia)^2             ▷ Eleva al cuadrado la distancia
9:             suma ← suma + potencia                ▷ Añade la diferencia a la suma
10:        end for
11:        distancia ← √suma                         ▷ Calcula la raíz cuadrada
12:        matrixEuclidean[i, j] ← distancia         ▷ Similaridad en la posición (i, j)
13:    end for
14: end for
15: return matrixEuclidean                           ▷ Retorna el matriz de afinidad
```

Figura 7. Algoritmo de cálculo de distancia euclidiana.

De las dos métricas añadidas anteriormente se ha encontrado que matemáticamente podrían estimarse partiendo de un mismo procedimiento, donde ajustando los parámetros de potencia usados, se logra obtener una expresión general.

Algoritmo 3 Matriz de distancias manhattan

```
1: m, n ← x.shape()
2: matrixManhattan ← [empty − array, shape(m, n)]
3: for i ← 0, i++, i < m do                          ▷ Iterando sobre cada muestra de prueba
4:     for j ← 0, j++, j < m do                      ▷ Iterando sobre el resto de muestras
5:         suma, diferencia, absoluto ← 0            ▷ (Re)inicia las variables
6:         for k ← 0, k++, k < n do                  ▷ Iterando sobre cada atributo
7:             diferencia ← x[i, k] − x[j, k]        ▷ Calcula la diferencia
8:             absoluto ← abs|diferencia|            ▷ Calcula el valor absoluto
9:             suma ← suma + absoluto                ▷ Añade la diferencia a la suma
10:        end for
11:        matrixManhaattan[i, j] ← suma             ▷ Similaridad en la posición (i, j)
12:    end for
13: end for
14: return matrixManhattan                           ▷ Retorna el matriz de afinidad
```

Figura 8. Algoritmo del cálculo de distancias manhattan.

Esta expresión general ha sido definida previamente y para lo cual en la ecuación (44) se ha planteado la relación de las variables desde las distancias euclidiana y manahattan hacia la minowski, donde p representa el factor de potencia el cual caracteriza el término de la derecha, cuyo valor al ser 1 representa la distancia manhattan, al ser 2 representa la distancia euclidiana, es decir, $p = 1,2,3,4,5,\ldots$

$$\left.\begin{aligned} d_E(x_i, x_j) &= \sqrt{\sum_{k=1}^{n} \left(x_{ik} - x_{jk}\right)^2} \\ d_T(x_i, x_j) &= \sum_{k=1}^{n} |x_{ik} - x_{jk}| \end{aligned}\right\} \rightarrow d_P(x_i, x_j) = \sum_{k=1}^{n} \left(|x_{ik} - x_{jk}|^p\right)^{1/p} \tag{44}$$

Teniendo clara la relación de las distancias, se ha obtenido el Algoritmo 4 en base a los Algoritmos 2 y 3, realizando la interpretación desde la ecuación (44).

Algoritmo 4 Matriz de distancias minkowski

```
m, n ← x.shape()
matrixMinkowski ← [empty − array, shape(m, n)]
for i ← 0, i++, i < m do                          ▷ Iterando sobre cada muestra de prueba
    for j ← 0, j++, j < m do                      ▷ Iterando sobre el resto de muestras
        suma, diferencia, potencia, distancia ← 0 ▷ (Re)inicia las variables
        for k ← 0, k++, k < n do                  ▷ Iterando sobre cada atributo
            diferencia ← abs|x[i, k] − x[j, k]|   ▷ Calcula la diferencia absoluta
            potencia ← (diferencia)^p             ▷ Eleva a la potencia p
            suma ← suma + potencia                ▷ Añade la diferencia a la suma
        end for
        distancia ← (suma)^(1/p)                  ▷ Calcula la raíz p
        matrixMinkowski[i, j] ← distancia         ▷ Similaridad en la posición (i, j)
    end for
end for
return matrixMinkowski                            ▷ Retorna el matriz de afinidad
```

Figura 9. Algoritmo de cálculo de distancias minkowsski.

Sin embargo, la literatura ha presentado un tipo de distancia que asume dentro de sus características la variabilidad de las muestras con respecto a su media, la cual es conocida como

distancia mahalanobis. En el Algoritmo 5 se ha sintetizado el procedimiento para obtener la matriz de distancias usando está métrica, la cual usa la matriz inversa de covarianza, aplicándola en un producto punto con la diferencia Δ entre las muestras y su respectivo Δ^T (ver ecuación (45)), de esta manera se itera sobre cada una de las muestras de manera similar a los Algoritmos 2-4, con el objeto de construir cada uno de los puntos de la matriz $m \times m$.

$$d_m = \sqrt{(x_i - x_j)\Sigma^{-1}(x_i - x_j)^T} \tag{45}$$

Por otro lado, el proceso de obtener la matriz de covarianza ha sido sintetizado en la sección anterior matemáticamente como el producto de la matriz de interés con su transpuesta, dicho esto, en el Algoritmo 5 se ha propuesto de esta manera en la línea 3, donde la inversa de una matriz es otro procedimiento algebraico el cual se debe evaluar como una función incluida en el toolbox del lenguaje, evitando recursos duplicados y optimizando el cálculo de la distancia.

Algoritmo 5 Matriz de distancias mahalanobis

```
1: m, n ← x.shape()
2: matrixMahalanobis ← [empty − array, shape(m, n)]
3: Σ = x · x^T                                          ▷ Matriz d Covarianza
4: for i ← 0, i++, i < m do                             ▷ Iterando sobre cada muestra
5:     for j ← 0, j++, j < m do                         ▷ Iterando sobre contra el resto de muestras
6:         sum ← 0,    Δ ← [0 ···]                   ▷ (Re)inicia las variables de suma y distancia
7:         for k ← 0, k++, k < n do                     ▷ Iterando sobre cada atributo
8:             Δ[k] ← x[i, k] − x[j, k]              ▷ Calcula la diferencia para cada atributo
9:         end for
10:        matrixMahalanobis[i, j] ← √(Δ · Σ^−1 · Δ^T)   ▷ Evalúa la ecuación (45)
11:    end for
12: end for
13: return matrixMahalanobis                            ▷ Retorna el matriz de afinidad
```

Figura 10. Algoritmo de cálculo de distancia mahalanobis.

3.2.3.2. Preprocesamiento y filtrado.

Por consiguiente, debido a las características específicas para cada tipo de variable, estás generalmente no presentan un formato estandarizado para operarse matemáticamente asumiendo

el peso de cada muestra sobre el resto, por tal se presenta en el Algoritmo 6 el normalizado intrínseco de las similaridades obtenidas en la matriz de afinidad, usando los valores máximos y mínimos para obtener el alcance de la muestra Δ_x y cada una de la muestras es referida sobre el resto usando el valor mínimo dentro de la matriz de afinidad, dando lugar a Δ_y, finalmente se realiza la división de las muestras referenciadas sobre el valor del alcance (valor moderador).

Algoritmo 6 Normalizado de la matriz de afinidad

$\Delta_x = |\mathrm{MAX}\{M\} - \mathrm{MIN}\{M\}|$ ▷ Obteniendo el alcance máximo entre las muestras
$\Delta_y = M - \mathrm{MIN}\{M\}$ ▷ Referenciando todos los valores con respecto al mínimo
$M_{NORM} = \frac{\Delta_y}{\Delta_x}$ ▷ Normalizando basado en el rango máximo y valores referenciados

Figura 11. Algoritmo de cálculo de la matriz de afinidad.

Luego de obtener esto y con el objeto de procesar los datos usando visión computacional, ha sido necesario modular los valores usando un valor mínimo y un valor máximo, siendo para el Algoritmo 7 el valor mínimo $s_{min} = 0$ y el valor máximo $s_{max} = 1$, este escalado es relativo al procesamiento de imágenes en R, puesto que para python se ha encontrado que con OpenCV se modifica el valor s_{max} a 255.

Algoritmo 7 Ajuste de escala

$s_{min} = 0$ ▷ Definición del valor mínimo
$s_{max} = 1$ ▷ Definición del valor máximo
$M_S = s_{min} + s_{max} * M_{NORM}$ ▷ Ajuste aplicado sobre la matriz normalizada

Figura 12. Algoritmo de escalado.

El procedimiento de escalado debe ser aplicado justo después del normalizado, con el objeto de obtener el formato deseado para la imagen, de lo contrario se ha encontrado que el evitar el normalizado de los datos evita la identificación correcta de las afinidades entre grupos y generando una matriz de datos con comportamientos indeseables.

3.2.3.3. Operaciones de visión computacional.

Para esta sección se ha preparado las muestras de datos y su matriz de afinidad para la identificación de los grupos, sin embargo, se deben realizar las consideraciones de homogenización del ruido, binarización y apertura de píxeles para eliminar regiones con información irrelevante en la identificación de los k-grupos. Inicialmente se ha propuesto la difuminación de los píxeles usando una ventana basada en el tamaño de la imagen de entrada (matriz de distancias), siendo para este propósito de tipo cuadrada, es decir, posee el mismo número de columnas al de filas. En el Algoritmo 8 se presenta los procedimientos planteados para la difuminación y homogenización del ruido, donde una venta ω planteada al 2% de la cantidad de muestras l^2 de la matriz de similaridad m, con los se estima la ventana $kernel$ y aplicada en el conjunto de datos estructurado $imageM$ a través de la convolución de matrices y dando como resultado final $imageBLUR$.

Algoritmo 8 Difuminado de la matriz de Similaridad

$l \leftarrow length(m)$
$w \leftarrow l * 0{,}02$ ▷ Correspondiente a obtener el 2 %
$\omega \leftarrow \text{Integer}\{w\}$
$kernel \leftarrow matrix(shape = (\omega, \omega))$
$kernel \leftarrow kernel/\omega^2$
$imageM \leftarrow reshape(m, (\omega, (l^2)/\omega))$
for $i \leftarrow 0, i = i + \omega, i = l$ **do**
    $imageBLUR = kernel * image[i + \omega^2]$
end for
$imageBLUR \leftarrow reshape(imageBLUR, (l, l))$

Figura 13. Algoritmo de difuminado de matriz de similaridad.

Seguidamente de esto, se procede a binarizar la imagen bajo el umbral c_B, el cuál es un criterio obtenido en 192 de los experimentos preliminares realizados sobre 100 muestras de datos, estando definido éste dentro de una escala de $\{0 - 255\}$, según este parámetro, un pixel

específico asume el valor 0 al ser inferior que c_B y un valor de 255 de ser igual o mayor al criterio.

Algoritmo 9 Binarización

```
l ← length(imageBLUR)
imageThresh ← imageBLUR.copy()
c_B ← 192
for i = 0, i++, i = l do
    for j = 0, j++, j = l do
        if imageThresh[i, j] < c_B then
            imageThresh[i, j] ← 0
        else
            imageThresh[i, j] ← 255
        end if
    end for
end for
```

Figura 14. Algoritmo de binarización.

Continuando con las operaciones morfológicas, se ha tomado la ecuación (34) dentro del Algoritmo 10, teniendo en cuenta que los lenguajes de programación R y Python, cuentan con librerías como “imager” y “openCV” con los operadores de erosión y dilatación optimizados y perfeccionados sobre imágenes, tanto binarizadas, como en otros formatos. En el Algoritmo 10 se evidencia el uso de un $kernel$ adicional, el cual ha sido determinado experimentalmente bajo 20 pruebas, determinando su valor en 3% de la cantidad total de muestras, del cual los valores superiores o inferiores a este demuestran un incremento en el error, asimilando el comportamiento de una campana de Gauss.

Algoritmo 10 Apertura y cierre de pixel

$imageOpenning \leftarrow imageThresh.copy()$
$l \leftarrow length(imageThresh)$
$w \leftarrow l * 0{,}03$ ▷ Correspondiente a obtener el 2 %
$\omega \leftarrow \text{Integer}\{w\}$
$kernel \leftarrow matrix(shape = (\omega, \omega))$
$kernel \leftarrow kernel/\omega^2$
$imageOpenning \leftarrow \text{erosion}\{imageOpenning.copy(), kernel\}$
$imageOpenning \leftarrow \text{dilatation}\{imageOpenning.copy(), kernel\}$

Figura 15. Algoritmo de apertura de pixeles.

Finalmente, después de todo el procesamiento usando operadores morfológicos, se ha usado el Algoritmo Canny ilustrado en la Figura 16,el cuál fue extraído usando los diferentes enfoques de , en los cuáles se encuentra el Operador Sobel mejorado a través de la segunda homogenización de los valores en los pixeles, para seguidamente obtener gradientes de cambio en direcciones transversales, continuando con la supresión del ruido y desarrollo de la histéresis, donde los bordes son evidentes para los objetos contenidos dentro de la imagen.

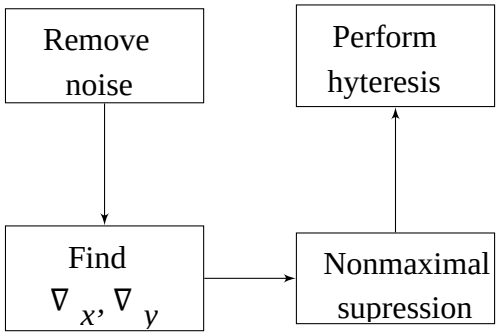


Figura 16. Diagrama de operaciones del algoritmo Canny.

3.2.3.4. Proyección lineal en un nuevo espacio.

Por último, en el Algoritmo 11 se ha estructurado la corrección de la superposición de grupos identificados, con el incremento de la distancia entre grupos y la reducción de la distancia entre muestras de un mismo grupo, de manera que la ecuación (1) es validada en las líneas 17 –

24, usando la máxima propagación y dispersión de muestras, tanto como las propiedades intrínsecas de estás, como su sentido y ángulo.

3.3. Desarrollo

Con los lenguajes de programación R y Python se ha encontrado gran variedad de librerías para propósitos de análisis de datos y aprendizaje de máquina, por lo cual la elección de instrumentos ha representado una viabilidad metodológica bastante ágil para tratamiento de datos como lo expresa (Bressert, 2012). En la Tabla 5 se presenta la utilidad de las librerías de R con respecto al objeto del Cluster CV2, asumiendo las áreas en que se ha desarrollado el trabajo.

Algoritmo 11 Algoritmo de proyección lineal en nuevo espacio [45]

$D_F \leftarrow D_\ell[].copy$ ▷ Crear nuevo marco de datos
for $g_k \leftarrow 1$, until k **do** ▷ Iterar en cada k-grupo identificado
  $[\mathbf{D}_k, C_{ik}, C_{jk}] \leftarrow [x_{ik}, x_{jk}, idx_k]$ ▷ Extract data
  $C_{ik} \leftarrow \frac{1}{n}\sum x_{ik}$
  $C_{jk} \leftarrow \frac{1}{n}\sum x_{jk}$
  procedure COVARIANCE MATRIX($\mathbf{D}_k, C_{jk}, C_{ik}$)
    $\mathbf{\Sigma_k} \leftarrow [\mathbf{D}_k\mathbf{D}_k^T, C_{jk}, C_{ik}]$
  end procedure
  procedure EIGEN-VALUES AND EIGEN-VECTORS($\mathbf{\Sigma}_k$)
    $(\overrightarrow{v}_k, \lambda_k) \leftarrow \Sigma_k \overrightarrow{v}_k = \lambda_k \overrightarrow{v}_k$ ▷ Covariance matrix
  end procedure
  $S = \begin{bmatrix} s_{x_i} & 0 \\ 0 & s_{x_j} \end{bmatrix} \quad R = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$
  procedure TRANSLATION($\mathbf{D}_k, R, S$)
    $(T_{ik}, T_{jk}) \leftarrow T = RS \rightarrow (\mathbf{D}_k)$
    $\mathbf{D}_k \leftarrow \mathbf{D}_k + T$ ▷ $\downarrow d_{\text{inter-samples}}$
  end procedure
  procedure REPULSION(r_{x_i}, r_{x_j})
    if $r_{x_i} \neq 0$ **then**
      $r_{x_i} \leftarrow -q * r_{x_i}$
    end if
    if $r_{x_j} \neq 0$ **then**
      $r_{x_j} \leftarrow -q * r_{x_j}$
    end if
    $\mathbf{D}_k \leftarrow \mathbf{D}_k + r$ ▷ $\uparrow d_{\text{inter-clusters}}$
  end procedure
  $D_f(k) = D_k$
end for

Figura 17. Algoritmo de corrección de la superposición.

Tabla 5. Librerías de R usadas en el Cluster CV2.

Nombre	Operaciones matriciales	Tratamiento de imágenes	Visualización de datos	Machine learning
imager	X		X	
dplyr	X			
squash		X	X	
purrr	X			
seriation	X		X	
clusterCrit				X
mlbench			X	X
cluster				X
e1071	X	X		X

De la misma manera, en la Tabla 6 se presentan las librerías para el lenguaje Python con sus funcionalidades para las áreas manejadas en el proyecto.

Tabla 6. Librerías de Python usadas en el Cluster CV2.

Nombre	Operaciones matriciales	Tratamiento de imágenes	Visualización de datos	Machine learning
numpy	X			
matplotlib			X	
pandas			X	
opencv		X	X	X
seaborn			X	
imageio		X	X	
scikit	X			X
sklearn	X			X
skimage	X	X	X	X

Con las herramientas de software claramente delimitadas, se ha procedido a la codificación de los algoritmos planteados en la sección anterior, haciendo uso de las herramientas predispuestas en las Tablas 5 y 6. Adicionalmente, en el trabajo de se ha realizado la implementación para el lenguaje R, por lo cual en este trabajo se ha usado la implementación de los autores en dicho lenguaje, replicándola y traduciéndola a lenguaje Python con el objeto de mejorar los resultados con respecto a precisión y velocidad de procesamiento, es decir, optimizarlo para ambos lenguajes.

3.3.1. Codificar algoritmo.

En la codificación se ha optado por el paradigma de programación procedimental, el cual engloba todos los procesos repetitivos dentro de métodos o funciones que son llamadas desde una estructura de archivos, esto con el objeto de simplificar el código y sea interpretable en lenguaje natural, en el caso de que la persona que lo lee tenga pleno manejo del inglés (Contreras Contreras, Medina Delgado, Guevara Ibarra, Leite de Castro, & Acevedo Jaimes, 2019).

3.3.1.1. Matriz de distancias.

En el código original del Cluster CV se encontraba el siguiente método para la obtención de la matriz de distancias, donde el propósito del Cluster CV2 es abordar 4 distancias ampliamente usadas, sin embargo, en su método solo se encuentran dos, donde el abordaje ha asumido los datos de manera estrictamente espacial, quedando el siguiente código en R:

```
# Function computing distances matrix
make.distancia <- function(xin, N, method="euclidian"){
 dis<-as.matrix(dist(xin, method="euclidian"))
 old.par <- par(mfrow=c(1, 1))
 image(seq(1,N,1),seq(1,N,1),dis, col = gray.colors(255, start = 0.1, end = 0.9,
 gamma = 2.2, alpha = NULL),
 main="Affinity matrix")
 par(old.par)
 return(dis)
}
```

Figura 18. Código para el cálculo de matrices de distancias en R.

Por lo tanto, en la implementación del algoritmo en Python, se ha codificado los Algoritmos del 2 al 5, con el uso de métodos disponibles en las librerías de python.

```
def makeDistance(xin,method): #
 ''' Similarity matrix computing function'''
 if method == 'euclidean' or 'L2':
 dis = makeDistanceE(xin)
 elif method == 'manhattan' or 'L1':
 dis = makeDistanceT(xin)
 elif method == 'minkowski' or 'L5':
 dis = makeDistanceP(xin,5)
 elif method == 'mahalanobis' or 'LSd':
 dis = makeDistanceM(xin)
 plt.imshow(dis,cmap='gray')
 plt.xlabel('$x_i$')
 plt.ylabel('$x_j$')
 plt.savefig('SM.eps')
 plt.savefig('SM.png')
 plt.show()
 print("Valores para la matriz de distancias M")
 print(".....")
 print("MAX(M): ",np.max(dis))
 print("MIN(M): ",np.min(dis))
 print("MEAN(M): ",np.mean(dis))
 print(".....")
 return dis
```

Figura 19. Código para el cálculo de matrices de distancias en Python.

La única matriz de distancias que no estaba definida previamente en la literatura ha sido la matriz de distancias mahalanobis, puesto que la librería scipy presentaba la implementación de la métrica, esta no lo ha abordado a nivel de matriz de distancias, por lo cual se ha desarrollado el método quedando el siguiente trecho de código:

```
def makeDistanceM(xin):
 ''' Mahalanobis computing function'''
 print(''' Mahalanobis computing function''')
 V = np.cov(xin.T)
 '''VI=np.invert(V) seems not be invese matrix computing, check equation'''
 VI = np.linalg.inv(V)
 m, n = xin.shape
 result = np.empty((m,m),dtype=float)
 for i in range(len(xin)):
  for j in range(len(xin)):
   result[i,j] = mahalanobis(xin[i,:],xin[j,:],VI)
 return result
```

Figura 20. Código de la distancia mahalanobis.

3.3.1.2. Matriz Normalizada.

La segunda función aborda la transformación de escala de los valores contenidos en la matriz de distancias, intentando identificar la variación de las atribuciones en escala de gris y proyectar lo en la imagen, siendo para R el código:

```r
# Function that scales affinity matrix to [0-1] span
make.normalizacao <-function(dis){
 smin=0; smax=1
 dis2<-( dis - min(dis) ) * smax / ( max(dis) - min(dis) ) + smin
 dis3<-1 - dis2 # seria la inversa, con 255 como mayor valor
 old.par <- par(mfrow=c(1, 2))
 image(seq(1,N,1),seq(1,N,1),dis2, col = gray.colors(255, start = 0.1, end = 0.9,
 gamma = 2.2, alpha = NULL),
 main="M. normal")
 image(seq(1,N,1),seq(1,N,1),dis3, col = gray.colors(255, start = 0.1, end = 0.9,
 gamma = 2.2, alpha = NULL),
 main="M. complemento")
 par(old.par)
 return(dis2)
}
```

Figura 21. Código de la función de normalizado en R.

Y para python, las asignaciones son casi idénticas a R, debido a que solo son operaciones matriciales definidas en la ecuación (27), quedando el siguiente código:

```python
def makeNorm(dis):
 smin = 0
 smax = 1
 dis2 = ((dis - np.min(dis))*smax)/(np.max(dis)-np.min(dis)) + smin
 return dis2
```

Figura 22. Código de la función de normalizado en Python.

3.3.1.3. Matriz filtrada.

Continuando, de la ecuación (29) se ha filtrado una serie de similaridades usando como referencia la variabilidad de los datos representa en la métrica de desviación estándar s_d nombrada como “limiar” en el siguiente trecho de código en R:

```r
# Function thresholder using standard deviation
make.limiar <- function(dis, limiar){
 media<-mean(dis)
 dp<-sd(dis)
 M2 <- dis
 #M2 [dis2 < dp] <- 1
 M2 [dis >= limiar*dp] <- 1
 #diag(M2)<-dp
 old.par <- par(mfrow=c(1, 1))
 image(seq(1,N,1),seq(1,N,1),M2, col = gray.colors(255, start = 0.1, end = 0.9,
 gamma = 2.2, alpha = NULL))
 par(old.par)
 return(M2)
}
```

Figura 23. Código de límite estadístico en R.

Según se ha visto en el código de arriba, basta con identificar cuales elementos presenta una variabilidad superior a la definida por el limiar, para R este limiar ha trabajado eficientemente en aproximadamente 1.3 y para Python, los valores varían desde 0.9 hasta 1.5 veces, dependiendo de la cantidad de distribuciones normales encontradas en los datos. El código respectivo a Python queda de la siguiente manera:

```python
def makeThresh(dis,limiar):
 dp = np.std(dis)
 M2 = dis
 M2[dis >= limiar*dp] = 1
 plt.imshow(M2,cmap='gray')
 imageio.imwrite('M2.png',M2)
 return(M2)
```

Figura 24. Código de límite estadístico en Python.

3.3.1.4. Procedimiento de etiquetado.

En esta sección se ha discutido el uso de una función de identificación de llamado: “split connected” en el Cluster CV, lo cual la metodología planteada en este trabajo designaba al algoritmo Canny de la Figura 16 como el identificador de los grupos, sin embargo, el método de mencionado previamente identifica efectivamente los grupos en R y teniendo en cuenta las operaciones morfológicas previas, el código ha quedado de la siguiente manera:

```
# Function computing distances matrix
# Funtion identifying k - clusters
make.metod.img <- function(imagem){
 file1_blur<-isoblur(matriz,1.6)
 plot(file1_blur, main = "borramento")
 im.t<-threshold(file1_blur,"25%")
 pxz <-im.t
 plot(pxz, main = "limiar")
 pxz1<-shrink(pxz,2) %>% grow(3)
 pxz2<-shrink(pxz1,1)
 pxz2<-as.pixset(pxz2)
 ff<-fill(pxz2,5) %>% clean(90)
 plot(pxz2, main = "detecao de clusters 1")
 highlight(ff,lwd=3)
 spxx <- split_connected(ff)
 a<-highlight(ff,lwd=3)

 prob<-fill(im.t,5) %>% clean(100)
 plot(im.t, main = "detecao de clusters 2")
 highlight(prob,lwd=3)

 file1.x<-as.pixset(1-matriz)
 plot(file1.x, main = "detecao de clusters 3")
 ppp<-fill(file1.x,5) %>% clean(120) #(5,120) para dados gaussianos 5 grupos
 plot(file1.x)
 highlight(ppp,lwd=3)

 sp <- split_connected(ppp) #returns an imlist
 numero.clusters<-length(sp)
 print(numero.clusters)
 plot(ppp)
 boundary(ppp) %>% where %$% { points(x,y,cex=.1,col="red") }
 return(list(sp=sp, numero.clusters=numero.clusters))
 }
```

Figura 25. Código del procedimiento de visión computacional en R.

Por otro lado, para python no ha sido encontrada la equivalencia del método “split connected” de R, lo cual el algoritmo canny ha predominado como el detector por defecto, este ha sido comparado con respecto al operador Sobel, pero este último no lograba homogenizar los bordes de un solo objeto. El algoritmo Canny con respecto al método “split connected” presenta criterios más estrictos de identificación, para lo cual ha sido necesario ampliar significativamente las ventanas de los operadores morfológicos, el código en python queda de la siguiente manera:

```
def makeMethodImg99():
 img=cv2.imread('M2.png')
 plt.imshow(img)
 plt.title('Visual information.')
 plt.savefig('img_0_image.svg')
 plt.show()

 long , channels, dimensions = img.shape
 criteriaCVblur = 0.02 #20x20/400
 W_est = int(criteriaCVblur*long)

 kernel = np.ones((W_est,W_est),np.uint8)/W_est
 file1_blur = cv2.filter2D(img,-1,kernel)
 plt.imshow(file1_blur)
 plt.title('Blurring.')
 plt.xlabel('$x_i$')
 plt.ylabel('$x_j$')
 plt.savefig('img_1_blur.svg')
 plt.show()

 criteriaCVthresh = 192 #[1-255]

 #im_t = cv2.THRESH_BINARY(file1_blur,63)
 im_t = cv2.threshold(file1_blur,criteriaCVthresh,255,cv2.THRESH_BINARY)
 #im_t = cv2.threshold(file2_blur,128,255,cv2.THRESH_TRUNC)
 pxz =  im_t.copy()
 plt.imshow(pxz[1])
 plt.title('Threshod binary.')
 plt.xlabel('$x_i$')
 plt.ylabel('$x_j$')
 plt.savefig('img_2_thresh.svg')
 plt.show()

 ''' Using : .....'''
 criteriaCVopen3x3 = 0.003 # 3x3/9
 kernel_3x3 = int(criteriaCVopen3x3*long)
 #Another criteria
 criteriaCVopen3x3 = 0.003 # 3x3/9
 kernel_3x3 = int(criteriaCVopen3x3*long)
 criteriaCVopen5x5 = 0.005 # 5x5/25
 kernel_5x5 = int(criteriaCVopen5x5*long)
```

Figura 26. Código del procedimiento de visión computacional en Python (1 de 4).

```
 criteriaCVopen20x20 = 0.005 # 20x20/400
 kernel_20x20 = int(criteriaCVopen20x20*long)
 criteriaCVopen25x25 = 0.03 # 5x5/25
 kernel_25x25 = int(criteriaCVopen25x25*long)

 kernelop = np.ones((kernel_20x20,kernel_20x20),np.uint8)/(kernel_20x20**2)
 opening2 = cv2.morphologyEx(pxz[1].copy(), cv2.MORPH_OPEN, kernelop)
 plt.imshow(opening2)
 plt.title('Morphological - Openning.')
 plt.xlabel('$x_i$')
```

Figura 27. Código del procedimiento de visión computacional en Python (2 de 4).

```
52    plt.ylabel('$x_j$')
53    plt.savefig('img_3_open.svg')
54    plt.show()
55
56    opb=opening2.copy()
57    kernel=np.ones((kernel_5x5,kernel_5x5),np.uint8)/(kernel_5x5**2)
58    opb=cv2.erode(opb,kernel,iterations=1)
59    kernel=np.ones((kernel_25x25,kernel_25x25),np.uint8)/(kernel_25x25**2)
60    opb=cv2.dilate(opb,kernel,iterations=1)
61    ##opb1 = cv2.cvtColor(opb, cv2.COLOR_BGR2GRAY)
62    opb2 = cv2.GaussianBlur(opb, (5,5), 1)
63    opb4 = cv2.Canny(opb2, 50, 200)
64    (_,opb4,_) = cv2.findContours(opb4.copy(), cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
65    cv2.drawContours(opb2,opb4,-1,(0,0,255),5)
66    plt.imshow(opb2)
67    plt.title('$k$ Number clusters.')
68    plt.xlabel('$x_i$')
69    plt.ylabel('$x_j$')
70    plt.savefig('img_4_detec.svg')
71
72    rotulos=np.arange(len(opb2))
73    centers = np.zeros(len(opb4))
74    for i in range(len(opb4)):
75     temp=np.repeat(1,(np.max(opb4[i])-np.min(opb4[i])))
76     centers[i]=np.mean(np.array([np.max(opb4[i]),np.min(opb4[i])]))
77     j=np.min(opb4[i])
78     while (j<=np.max(opb4[i])):
79      rotulos[j]=i+1
80      j=j+1
81    rotulos=np.where(rotulos<(len(opb4)+1),rotulos,len(opb4)+1)
82    unsigned = np.int_(np.where(rotulos==len(opb4)+1))
83    dist_comp = np.zeros((len(unsigned[0]),len(opb4)))
84    for i in range(len(unsigned[0])):
85     for j in range(len(opb4)):
86      dist_comp[i,j] = abs(unsigned[0,i]-centers[j])
87     rotulos[unsigned[0,i]]=np.int_(np.where(dist_comp[i]==np.min(dist_comp[i])))[0,0]+1
88
89    if len(opb4) < 5:
90     img_result = img.copy()
91     cv2.drawContours(img_result,opb4,-1,(255,0,0),5)
```

Figura 28. Código del procedimiento de visión computacional en Python (3 de 4).

```
92    colorsG=['yellow','mediumseagreen','darkslateblue','purple','steelblue']
93    tags =['1','2','3','4','5']
94    font = cv2.FONT_HERSHEY_TRIPLEX
95    for cntr in range(len(opb4)):
96     cv2.putText(img_result,tags[cntr],(int(np.mean(opb4[cntr])-30),int(np.mean(opb4[cntr])
       +30)),
97     font, 3, (0,0,255),7, cv2.LINE_AA)
98    plt.imshow(img_result)
99    plt.title('$k$ Number of clusters Identification.')
100   plt.xlabel('$x_i$')
101   plt.ylabel('$x_j$')
102   plt.savefig('img_5_result.svg')
103
104  if(np.min(opb4[0])<=np.max(opb4[1])):
105   rotulos = False
106  return np.array([rotulos,len(opb4)])
```

Figura 29. Código del procedimiento de visión computacional en Python (4 de 4).

3.3.1.5. Corrección de la superposición.

```
# Function correcting overlaping issue
make.PCA <- function(xin,datos.novo,numero.clusters){
centroide.gnral.x<-mean(xin[,1])
centroide.gnral.y<-mean(xin[,2])

array.final<-list()
for(ii in 1:numero.clusters){
dat1<-datos.novo[,1][which(datos.novo[,3] %in% ii)]
dat2<-datos.novo[,2][which(datos.novo[,3] %in% ii)]
dat3<-datos.novo[,3][which(datos.novo[,3] %in% ii)]
data1<-matrix(c(dat1,dat2,dat3),ncol = 3)
x1 <- data1[,1]
y1 <- data1[,2]
m <- matrix(c(x1,y1),ncol=2)
centroidesx<-mean(m[,1])
centroidesy<-mean(m[,2])
cov.m <- cov(m)
cov.m
cov.eig <- eigen(cov.m)
cov.eig

angulo.inclinacion.pc1<-(atan(cov.eig$vectors[1,1]/cov.eig$vectors[2,1]))*180/pi
angulo.inclinacion.pc1
angulo.inclinacion.pc2<-(atan(cov.eig$vectors[1,2]/cov.eig$vectors[2,2]))*180/pi
angulo.inclinacion.pc2

f.vector2 <- as.matrix(cov.eig$vectors[,c(1,2)],ncol=2)
f.vector2
final2 <- t(f.vector2) \% *\% t(m)
final2

ang<- (1*pi)/180
rotacion<-matrix(c(cos(ang),sin(ang),-sin(ang),cos(ang)),nrow = 2,ncol = 2)
rotacion
sx<-0.9
sy<-0.9
escala<- matrix(c(sx,0,0,sy),nrow = 2,ncol = 2)
escala
rotesc<- rotacion \% *\% escala
rotesc
```

Figura 30. Código de corrección de la superposición (1 de 3).

Finalmente, la corrección de la superposición se ha realizado con los procedimientos matemáticos del análisis de componentes principales, descrito en el Algoritmo 11 del artículo (Contreras Contreras, Medina Delgado, Guevara Ibarra, Leite de Castro, & Acevedo Jaimes, 2019), implementado para R de la manera presentada en las Figuras.

```
final4z<-(f.vector2) \% *\% t(rotesc)
final4z
final4<- t(final4z) \% *\% t(m)
final4
original.dataset2.mod2 <- t(f.vector2 \% *\% final4)
original.dataset2.mod2

centroides.modx<-mean(original.dataset2.mod2[,1])
centroides.mody<-mean(original.dataset2.mod2[,2])

traslacion.x<-(centroidesx-centroides.modx)
traslacion.y<-(centroidesy-centroides.mody)

original.dataset2.mod23<-matrix(0,ncol = 2,nrow = length(original.dataset2.mod2[,1]))
original.dataset2.mod23[,1]<-original.dataset2.mod2[,1] + traslacion.x
original.dataset2.mod23[,2]<-original.dataset2.mod2[,2] + traslacion.y

repulsion.x<-(centroide.gnral.x - (mean(original.dataset2.mod23[,1])))
repulsion.y<-(centroide.gnral.y -(mean(original.dataset2.mod23[,2])))

if(repulsion.x < 0) {
repulsion.xx<- -1.9*repulsion.x
}
if(repulsion.x > 0){
repulsion.xx<- -1.9*repulsion.x
}
```

Figura 31. Código de corrección de la superposición (2 de 3).

```

if(repulsion.y < 0) {
repulsion.yy<- -1.9*repulsion.y
}
if(repulsion.y > 0){
repulsion.yy<- -1.9*repulsion.y
}

original.dataset2.mod233<-matrix(0,ncol = 2,nrow = length(original.dataset2.mod2[,1]))
original.dataset2.mod233[,1]<-original.dataset2.mod23[,1] + repulsion.xx
original.dataset2.mod233[,2]<-original.dataset2.mod23[,2] + repulsion.yy
plot(original.dataset2.mod233[,1],original.dataset2.mod233[,2],xlim=c(-4,4),
ylim=c(-4,4),col=ii)
par(new=T)
data31<-matrix(c(original.dataset2.mod233[,1],original.dataset2.mod233[,2],dat3),
ncol = 3)
array.final[[ii]]<-data31

}
return(array.final)

}
```

Figura 32. Código de corrección de la superposición (3 de 3).

De la misma manera, usando los mismos procedimientos matriciales dentro del lenguaje python:

```
1
2  def PCA(datos_novo,n_clusters):
3
4   centroide_gnral_x = datos_novo[0].mean()
5   centroide_gnral_y = datos_novo[1].mean()
6   array_final = []
7    for i in (range(n_clusters)):
8   dat1 = np.extract(datos_novo[2]==(i+1),datos_novo[0])
9   dat2 = np.extract(datos_novo[2]==(i+1),datos_novo[1])
10  dat3 = np.extract(datos_novo[2]==(i+1),datos_novo[2])
11  data1 = np.transpose(np.array([dat1,dat2,dat3]))
12  x1 = dat1
13  y1 = dat2
14  #plt.scatter(x1,y1)
15  m = np.array([x1,y1])
16   centroidesx = np.mean(x1)
17   centroidesy = np.mean(y1)
18   cov_m = np.cov(m)
19   if len(cov_m) != 2:
20    array_final = False
21   else:
22    cov_eig = np.linalg.eig(cov_m)#[0]values,[1]vectors
23    cov_eig_values=cov_eig[0]
24    cov_eig_vectors=cov_eig[1]
25    cov_eig
26    #plt.scatter(x1,y1)
27
28    angulo_inclinacion_pc1 = (np.arctan(cov_eig_vectors[0,0]/cov_eig_vectors[1,0]))*180/math.
       pi
29    angulo_inclinacion_pc1
30    angulo_inclinacion_pc2 = (np.arctan(cov_eig_vectors[0,1]/cov_eig_vectors[1,1]))*180/math.
       pi
31    angulo_inclinacion_pc2
32
33    f_vector2 = cov_eig_vectors# feature vector with both components
34    f_vector  =  2
35    final2 = np.matmul(f_vector2,m) # new dataset for feature vector 2
36    final2
37
```

Figura 33. Codificación del análisis de componentes principales (1 de 2).

```
    ang = (math.pi)/180
    rotacion = np.array([[math.cos(ang),math.sin(ang)],[-math.sin(ang),math.cos(ang)]])
    rotacion
    sx = 0.6 #0.6
    sy = 0.6
    escala = np.array([[sx,0],[0,sy]])
    escala
    rotesc = np.matmul(rotacion,escala)
    rotesc
    final4z = np.matmul(np.transpose(f_vector2),np.transpose(rotesc))
    final4z
    final4 = np.matmul(np.transpose(final4z),m)
    final4
    original_dataset2_mod2 = np.matmul(f_vector2,final4)
    original_dataset2_mod2

    centroides_modx = np.mean(original_dataset2_mod2[0])
    centroides_mody = np.mean(original_dataset2_mod2[1])

    traslacion_x = (centroidesx - centroides_modx)
    traslacion_y = (centroidesy - centroides_mody)

    original_dataset2_mod23 = np.zeros((2,len(original_dataset2_mod2[0])))
    original_dataset2_mod23[0] = original_dataset2_mod2[0] + traslacion_x
    original_dataset2_mod23[1] = original_dataset2_mod2[1] + traslacion_y

    repulsion_x = (centroide_gnral_x - (np.mean(original_dataset2_mod23[0])))
    repulsion_y = (centroide_gnral_y -(np.mean(original_dataset2_mod23[1])))

    if(repulsion_x < 0):
     repulsion_xx = -2.4*repulsion_x
    if(repulsion_x > 0):
     repulsion_xx = -2.4*repulsion_x
    if(repulsion_y < 0):
     repulsion_yy = -2.4*repulsion_y
    if(repulsion_y > 0):
     repulsion_yy = -2.4*repulsion_y

    original_dataset2_mod233=np.zeros((2,len(original_dataset2_mod2[0])))
    original_dataset2_mod233[0] = original_dataset2_mod23[0] + repulsion_xx
    original_dataset2_mod233[1] = original_dataset2_mod23[1] + repulsion_yy
    plt.scatter(original_dataset2_mod233[0],original_dataset2_mod233[1])
    data31 = np.array([original_dataset2_mod233[0],original_dataset2_mod233[1],dat3])
    data31 = np.transpose(data31)
    array_final.append(data31)
return array_final
```

Figura 34. Codificación del análisis de componentes principales (2 de 2).

3.3.2. Pruebas unitarias.

Inicialmente se habían planteado las 4 distribuciones probabilísticas estadísticas de la Figura 35, donde cada una presenta un cambio de variabilidad distinto y determinado por su modelo general.

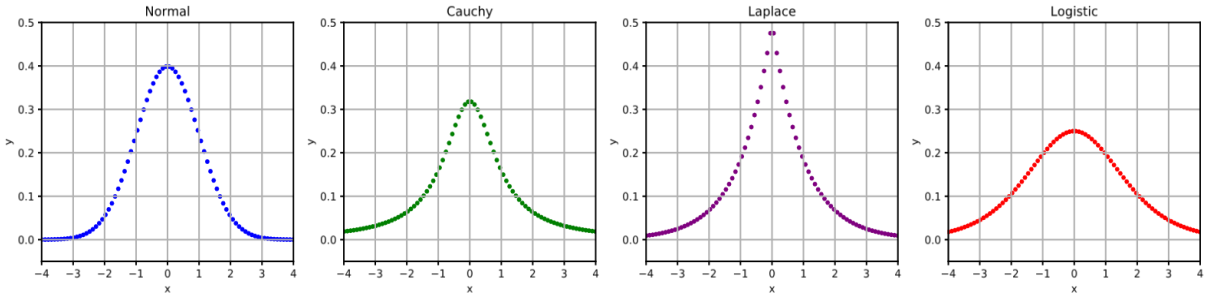


Figura 35. Distribuciones probabilistas.

Sin embargo, estas distribuciones han sido sometidas a dos pruebas para la implementación de las mismas dentro de la generación de datos artificiales del Cluster CV2, donde la que ha representado un comportamiento inusual ha ido la distribución Cauchy, según se presenta en la Figura 36, La distribución presenta muestras altamente separadas del centroide general, alcanzando valores en los cientos.

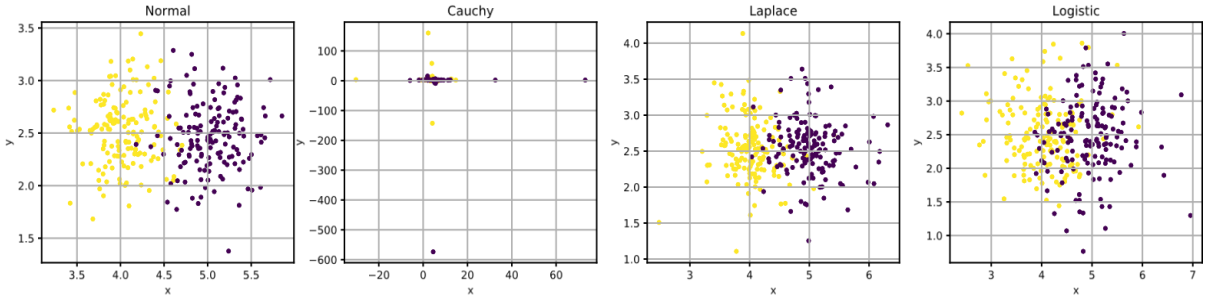


Figura 36. Distribución espacial de las muestras probabilistas.

Por lo tanto, para determinar la morfología de grupos, se ha presentado en la Figura 37 el histograma de valores, evidenciando la alta variabilidad de la distribución Cauchy y los dos grupos a encontrar en el resto de las distribuciones.

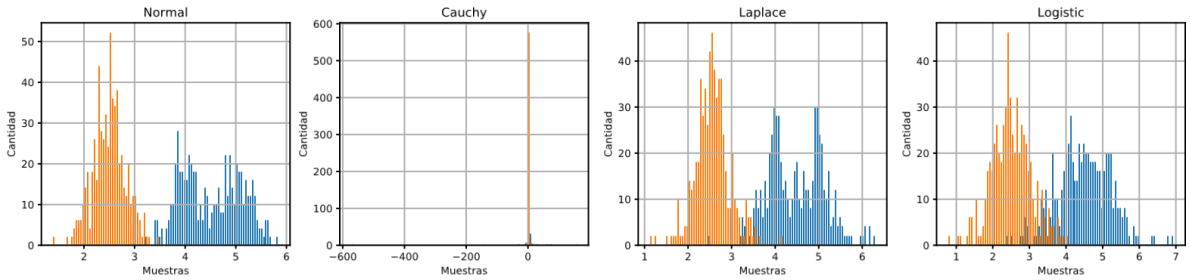


Figura 37. Histogramas de las distribuciones probabilísticas.

Adicionalmente se evidencia que las distribuciones Normal y Laplace presentan una tendencia a dividir las muestras de un grupo en dos, elevando la complejidad de la detección de las muestras, siendo la Figura 38 las matrices de distancias para cada una de las distribuciones, debido a que los valores son altamente variables.

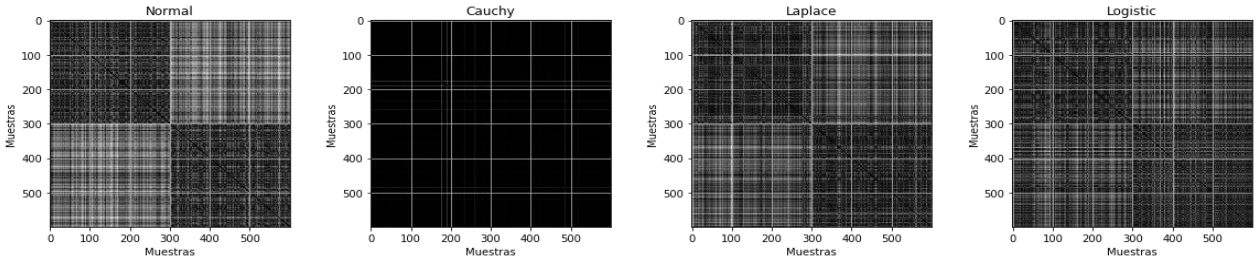


Figura 38. Matrices de distancias para las distribuciones probabilísticas.

De esta manera, cada una de las similaridades contenidas en la matriz de distancias es normalizada usando sus valores máximos y mínimos, tal como se presenta en la Figura 39, evidenciando un leve cambio, lo cual indica que a pesar de que se encuentra normalizada, la matriz sigue conservando alta variabilidad (Navidi, 2006).

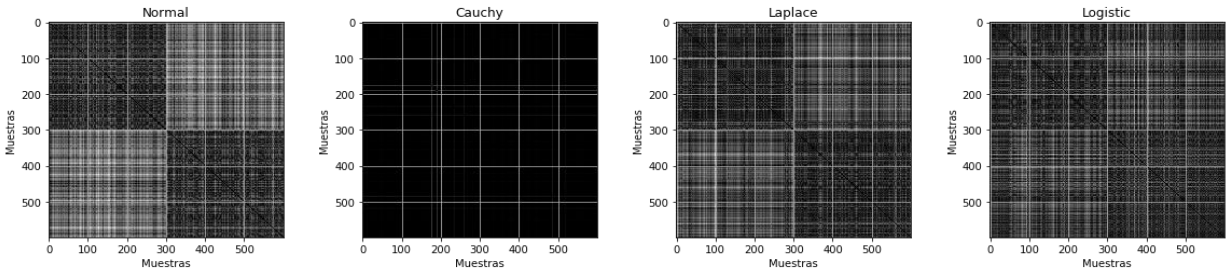


Figura 39. Matrices de distancias normalizadas.

Por consiguiente, se aplica el limiar propuesto por el trabajo de (Jaimes, Castro, Torres, Silva, & Braga, 2017), donde la matriz reduce su variabilidad significativamente (ver Figura 40) para lograr una debida identificación de los grupos. Adicionalmente, en la misma se identifica en las afinidades de la distribución Cauchy no evidencian claramente la presencia de grupos, pues como se había previsto desde a Figura 36, la distribución tiene sus muestras muy dispersas para un enfoque espacial.

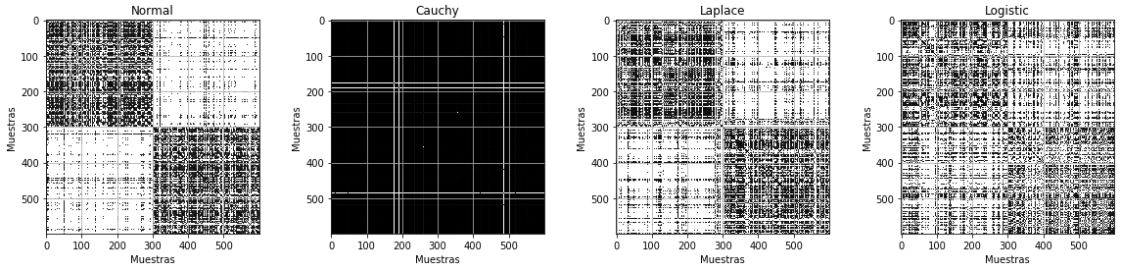


Figura 40. Matrices de distancias normalizadas y limiarizadas bajo el criterio del Cluster CV.

Finalmente, en la Figura 41 se presenta la identificación lograda para la distribución Normal y Laplace, donde se evidencia problemas previstos en la Figura 36, la cual evidenciaba la tendencia a generar 3 grupos.

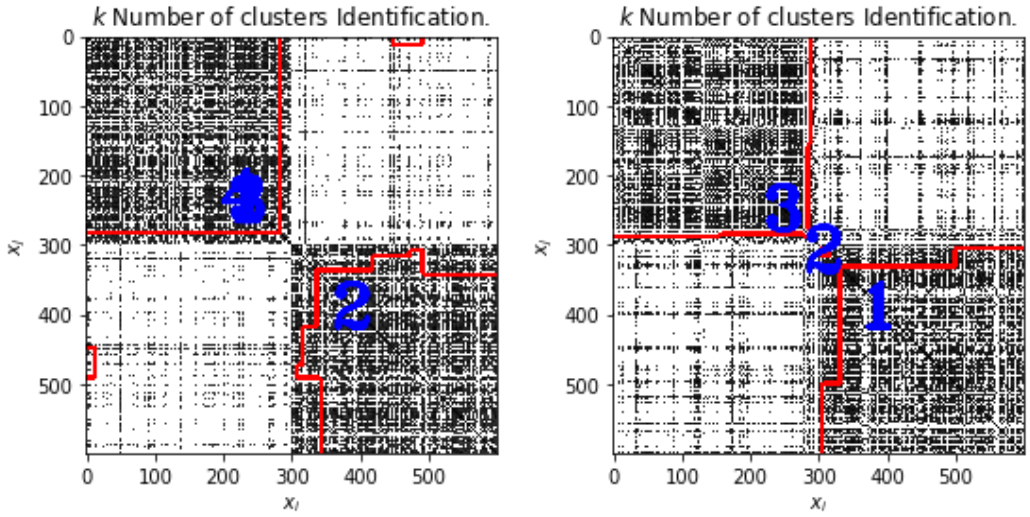


Figura 41. Identificación de k-grupos lograda.

3.3.3. Codificación de ayudas.

Con lo anterior, generalmente ha sido necesario usar métodos de ayuda que expliquen o dinamicen la aplicación del Cluster CV2 por el usuario, para lo cual se ha desarrollado un sistema de ayudas para las funciones generales de datos, donde la persona llama el método:

clusterCV2helper(‘makeDistacia’)

El cual regresa un texto ilustrando los procedimientos con el método definido, además de una lista de ejemplos, los cuales disponen de una manera sencilla de llamar y probar. En el caso de que no se encuentre la función especificada por el usuario, este dará algunas instrucciones de cómo usar el asistente, además de presentar las opciones básicas con este.

3.4. Pruebas de funcionamiento

Está sección está enfocado o evaluar el desempeño del abordaje usando datos sintéticos y reales, donde cada uno presenta un comportamiento diferente, dependiendo de las situaciones teóricas planteadas para cada una. Por esto, inicialmente se crean las rutinas de estructuración de datos sintéticos, con información a priori suficiente para evaluar la calidad del agrupamiento; seguidamente se presenta el modo de ejecución para datos reales, estos son más complicados, pues no se tiene información a priori acerca del grupo al cual debería ser atribuido.

3.4.1. Datos sintéticos.

En la sección de pruebas unitarias, se dio una breve demostración del comportamiento de un conjunto de datos generados de manera artificial, por lo cual se ha estructurado los métodos de creación de datos sintéticos de la siguiente manera:

```
def unbalancedData(s):
 nc2 = 800
 nc1 = 100
 xc1 = (s*np.repeat(np.random.normal(size=nc1),
  2))+ np.matrix.transpose(np.repeat(np.array([4,2.5]),
 nc1))
 xc1 = np.transpose(np.reshape(xc1,(2,100)))
 yc1 = np.repeat(2,nc1)
 xc5 = (s*np.repeat(np.random.normal(size=nc2),
  2)) + np.matrix.transpose(np.repeat(np.array([5,2.5]),
 nc2))
 xc5 = np.transpose(np.reshape(xc5,(2,800)))
 yc5 = (np.repeat(1,nc2))

 xin = np.concatenate((xc1,xc5))
 yin = np.concatenate((yc1,yc5))

 dataIN = pd.DataFrame(np.column_stack((xin,yin)))
 dataIN.to_csv('dataIN.txt',index=False,header=None)

 return [xin,yin]
```

Figura 42. Rutina para datos sintéticos desbalanceados.

Para un conjunto de muestras desbalanceadas de distribución Normal, donde se define un número fijo para la cantidad de muestras totales en cada grupo, siendo de esta manera etiquetada al mismo tiempo que se van generando. Seguramente el concepto de muestras desbalanceados ha sido uno de los interrogantes de la afirmación anterior, por lo cual, en Cluster CV2 se ha tomado el concepto de conjunto de datos con muestras desbalanceadas a aquel donde sus grupos no poseen aproximadamente la misma cantidad de muestras, siendo de esta manera creada por otro lado el método de muestras balanceadas, asumiendo 5 grupos con igual cantidad muestras contenidas para cada grupo:

```
def balancedData(s):
 nc = 180
 xc1 = s*(np.repeat(np.random.normal(size=nc),
  2)) + np.matrix.transpose(np.repeat(np.array([4,2.5]),
 nc))
 xc1 = np.transpose(np.reshape(xc1,(2,180)))
 yc1 = (np.repeat(5,nc))
 xc2 = (s*np.repeat(np.random.normal(size=nc),
  2)) + np.matrix.transpose(np.repeat(np.array([3,4]),
 nc))
 xc2 = np.transpose(np.reshape(xc2,(2,180)))
 yc2 = (np.repeat(4,nc))
 xc3 = (s*np.repeat(np.random.normal(size=nc),
  2)) + np.matrix.transpose(np.repeat(np.array([4.2,5]),
 nc))
 xc3 = np.transpose(np.reshape(xc3,(2,180)))
 yc3 = (np.repeat(3,nc))
 xc4 = (s*np.repeat(np.random.normal(size=nc),
  2)) + np.matrix.transpose(np.repeat(np.array([5,3.5]),
 nc))
```

Figura 43. Rutina de creación de datos balanceados (1 de 2).

```
 xc4 = np.transpose(np.reshape(xc4,(2,180)))
 yc4 = (np.repeat(2,nc))
 xc5 = (s*np.repeat(np.random.normal(size=nc),
  2)) + np.matrix.transpose(np.repeat(np.array([5,2.5]),
 nc))
 xc5 = np.transpose(np.reshape(xc1,(2,180)))
 yc5 = (np.repeat(1,nc))

 xin = np.concatenate((xc1,xc2,xc3,xc4,xc5))
 yin = np.concatenate((yc1,yc2,yc3,yc4,yc5))

 dataIN = pd.DataFrame(np.column_stack((xin,yin)))
 dataIN.to_csv('dataIN.txt',index=False,header=None)

 return [xin,yin]
```

Figura 44. Rutina de creación de datos balanceados (2 de 2).

Estas funciones se repiten para cada tipo de distribución, donde el cambio es determinado por el modelo de cada una de ellas, queda mencionar que la distribución Cauchy a representado

un caso crítico para el mismo agrupamiento, por lo que no ha sido considerada dentro de los generadores de datos sintéticos y reservado su análisis hasta la demostración presentada previamente en las pruebas unitarias.

3.4.2. Datos reales.

Para esta sección, se ha realizado una búsqueda de bases de datos con información cruda y procesada, donde generalmente, otros autores no presentan los datos crudos, en vez de esto, se presentan dichos datos procesados y de manera generalizada, en los casos donde han se dispuesto los datos para replicar el trabajo realizado por los mismos. Debido a esto, después de dos meses de búsqueda sin mayor resultado, se ha encontrado tres bases de datos bajo la disponibilidad de este proyecto: la primera consiste en 244 mediciones de campo eléctrico, magnético y potencia de radiación realizadas en el campus UFPS (Beltrán Ortega, Guevara Ibarra, & Mendoza García, 2018); el siguiente conjunto de datos fue adquirido en un trabajo de investigación presentado en el encuentro quinto internacional de investigadores en materiales y tecnología de plasma (Contreras Contreras, Dulcé-Moreno, & Ardila Melo, Arduino data-logger and artificial neural network to data analysis, 2019), consistiendo en datos de un sistema térmico para incubación de huevos de aves domésticas; la tercera base de datos, fue obtenida por un egresado de la UFPS, quien solicitó y obtuvo datos meteorológicos del IDEAM (Ortiz Diaz & Sepúlveda Mora, 2018); y finalmente, la última base de datos usada, consiste en las 19 imágenes satelitales facilitadas por el Laboratório de Inteligência e Tecnologia Computacional de la Universidade Federal de Minas Gerais. De las bases datos mencionadas previamente, se ha abordado la de radiación no ionizante puesto que es un trabajo basado en una campaña de mediciones con un análisis estadístico medio y el conjunto de datos para el sistema térmico, pues hasta este punto, se han logrado interesantes para cada uno. Los datos de variables meteorológicas del IDEAM e imágenes satelitales son

bastante amplios y con estudio estadístico bajo, además de requerir una extensa depuración de la información.

3.4.2.1. Radiación no ionizante.

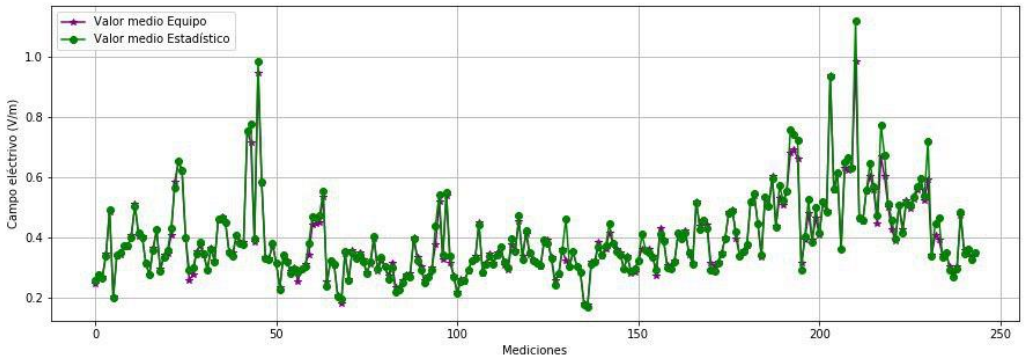


Figura 45. Mediciones de campo eléctrico medio.

El Grupo de Investigación y Desarrollo en Electrónica y Telecomunicaciones, mediante el desarrollo de un trabajo de grado en medidas de radiación no ionizante, dispuso las medidas obtenidas en el siguiente aplicativo web, donde han sido depuradas con respectos al rango de medición establecido de 0 a 20 V/m para campo eléctrico. Estás medidas fueron obtenidas en una campaña de mediciones realizadas en la Universidad Francisco de Paula Santander en el trabajo de grado (Beltrán Ortega, Guevara Ibarra, & Mendoza García, 2018), dando como resultado los valores de la Figura 45, con la media obtenida directamente desde el equipo y la calculada con los parámetros de variabilidad estadística almacenados en la base de datos.

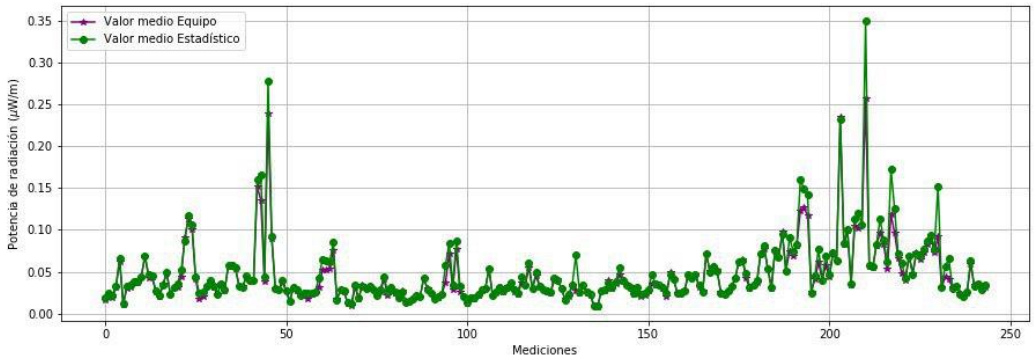


Figura 46. Mediciones de potencia de radiación media.

De esta manera el equipo arroja la medida de potencia de radiación (ver Figura 46), asumiendo que el medio es el aire, es decir, características isotópicas para campos electromagnéticos.

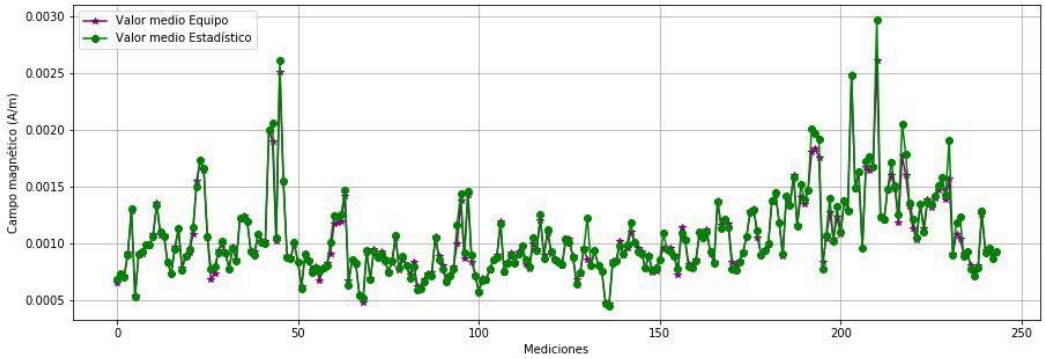


Figura 47. Mediciones de campo magnético medio.

Y de la misma manera se estima el campo magnético, completando las componentes del campo electromagnético y graficando el campo magnético en la Figura 47.

3.4.2.2. Sistema de adquisición de datos.

Sin embargo, obtener datos reales en la literatura ha sido una tarea laboriosa, aunque se consiguen gran cantidad de repositorios, la mayor parte de ellos no son de tipo exploratorios, pues son datos procesados y clasificados o para elaboración de clasificadores, entrenándolos con

entrada y salidas conocidas. Por consiguiente, en este trabajo se abordó la construcción de un sistema embebido de adquisición de datos usando la tarjeta de desarrollo arduino, financiado por el Semillero de Investigación UFPS-Foristom en Instrumentación y Materiales. El sistema final se presenta en la Figura 48, donde se elaboró el esquema apuntando a monitorear las variables termodinámicas de un sistema de incubación artificial de aves domésticas.

El desarrollo de este producto se deja en el Anexo C, donde se presenta las características de construcción, ensamble, metodología, valoraciones y aplicabilidad, como también el resumen general de este (Contreras Contreras, Dulcé-Moreno, & Ardila Melo, Arduino data-logger and artificial neural network to data analysis, 2019).

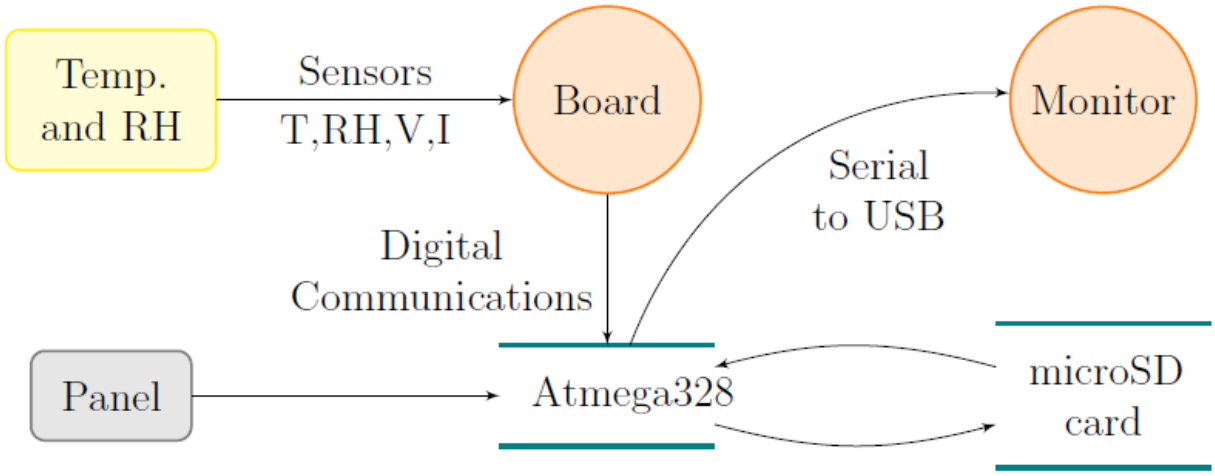


Figura 48. Flujo de datos para monitoreo de sistemas termodinámicos y variables ambientales.

3.4.3. Ejecución continua.

Finalmente, para evaluar efectivamente los costos computacionales del algoritmo, se ha elaborado la estructura modular presentada abajo, pues esta asegura la ejecución continua de pruebas usando muestras aleatorias y guardando la información relevante de las métricas de validación en el almacenamiento del entorno de ejecución:

Adicionalmente se almacenan y monitorea las variables externas al algoritmo, como los son el consumo promedio de memoria RAM y velocidad del procesador, donde la arquitectura modular aplicando el paradigma de programación procedimental asegura la reducción de estos costos al evitar variables globales y operando internamente en cada método.

```
import os
import pandas as pd
import time
from pro_clusterCV import desvalan
from pro_clusterCV import valan
import funciones
import artificialData
from validation import estadistica as criteria

def experiment1(sd,title,veces):
 print('Iniciado: ',time.strftime("%"))
 computer_path = os.environ['HOMEPATH']
 os.mkdir(computer_path+'\\GitHub\\Cluster-CV2\\artificial\\STSIVA2019\\'+title)
 os.chdir(computer_path+'\\GitHub\\Cluster-CV2\\artificial\\STSIVA2019\\'+title)
 #s = sd
 n_proba = 0
 n_correctas = 0
 l_correctas = []
 while n_correctas < veces :
  os.mkdir('proba'+str(n_proba))
  os.chdir('proba'+str(n_proba))
  print('Prueba: ',n_proba)
  acierto = valan(sd)
  if acierto:
   n_correctas += 1
   l_correctas.append(n_proba)
  if n_proba == veces*5:
  b reak
  print('Prueba ',n_proba,' completada')
  n_proba += 1
  os.chdir(computer_path+'\\GitHub\\Cluster-CV2\\STSIVA2019\\'+title)
 pd.DataFrame(l_correctas).to_csv('proba1.txt',index=False,
  header = None, sep=',')
 pd.DataFrame(l_correctas).to_csv('correctas.txt',
  index=False, header=None, sep=',')
 report=pd.DataFrame([['Fecha','No. pruebas','No. exitosas','No. fallidas'],
  [time.strftime("%"),n_proba,n_correctas,
  n_proba-n_correctas]])
 report.to_csv('report.txt',
  index=False, header=None, sep=',')
 print('Finalizado: ',time.strftime("%"))
 os.chdir(computer_path+'\\GitHub\\Cluster-CV2\\STSIVA2019\\')
return report
```

Figura 49. Sistema automático para pruebas masivas.

3.5. Entrega

Después de obtener el producto estable y funcional de la metodología, se ha preparado el mismo para su divulgación y adaptación a aplicaciones externas de este trabajo, por lo que esta

sección divide el procedimiento de entrega en tres etapas: la elaboración del manual, donde el usuario logre aplicar el Cluster-CV2 rápida y eficientemente sobre sus datos; seguido de la documentación del código con los herramientas propias de los lenguajes de programación; y finalmente, la distribución al público bajo licencia MIT en un repositorio en línea de GitHub.

3.5.1. Elaboración del manual.

Teniendo clara la cantidad de atribuciones diferentes para abordar una base de datos real, se ha elaborado una plantilla de ejecución para datos reales, donde el usuario debe variar o definir las variables que ha de someter a la metodología, debido a que teóricamente podría existir una Hipótesis previa o suposición de como las atribuciones se deben comportar. A continuación, se presenta el trecho de código de como procesar los datos de radiación no ionizante:

```
''' Checking what operative system is '''
computer_path = OSpath.OSpath()
route = computer_path+'/GitHub/Cluster-CV2'

''' changing directory '''
#os.chdir('/Users/Ghiordy F. Contreras/GitHub/Cluster-CV2/')
os.chdir(route)

''' Selecteing directory with data set '''
#os.chdir('/Users/Ghiordy F. Contreras/GitHub/ProjectBackup/data')

''' Read data set dependently from its format '''
```

Figura 50. Verificador de sistema operativo.

```
13 #data = pd.read_csv('maye.csv') # crossfit
14
15 ''' Organizing data with its features and instances '''
16 XIN = pd.read_csv('data/radiation-non-ionizant/radiationNONioOPEN.csv')
17 xin = XIN.values
18
19 ''' similarity metric used into a matrix '''
20 m = makeDistancia.makeDistanceM(xin[:,[3,6,9]])
21
22 ''' Carring to [0,1] range of values - Norming '''
23 n = makeNormalizacao.makeNorm(m)
24
25 ''' Math tresh using data variability '''
26 l = makeLimiar.makeThresh(n,1.3) #1.5
27
28 ''' Computer vision identification of k-clusters '''
29 result = makeLabel.makeCVlabeling('M2.png')
30 print(time.strftime("%c"),' was detected k-clusters: ',result[1])
31 plt.imshow(l,cmap='gray')
```

Figura 51. Selector de atribuciones objetivas.

En el cuál, se ha de revisar parámetros del entorno de ejecución como el sistema operativo, pues los datos reales son almacenados en formato comma separated value (CSV), el cual se encuentra en el almacenamiento interno del entorno y debe ser diseccionando correctamente desde el directorio raíz. Seguidamente, se seleccionan las atribuciones que se desean someter, indicándolas en la variable “xin”, el cual el trecho de código las almacena bajo el formato de matriz numpy; para finalmente aplicar la metodología desde la línea 19 a la 31.

3.5.2. Documentación del código.

Por consiguiente, se ha documentado el código en python usando los “docStrings” proporcionados por el mismo lenguaje de programación, un ejemplo se presenta a continuación:

```
1 ''' Checking what operative system is '''
2 computer_path = OSpath.OSpath()
3 route = computer_path+'/GitHub/Cluster-CV2'
4 ''' changing directory '''
5 #os.chdir('/Users/Ghiordy F. Contreras/GitHub/Cluster-CV2/')
6 os.chdir(route)
```

Figura 52. Ejemplo de documentación y comentarios.

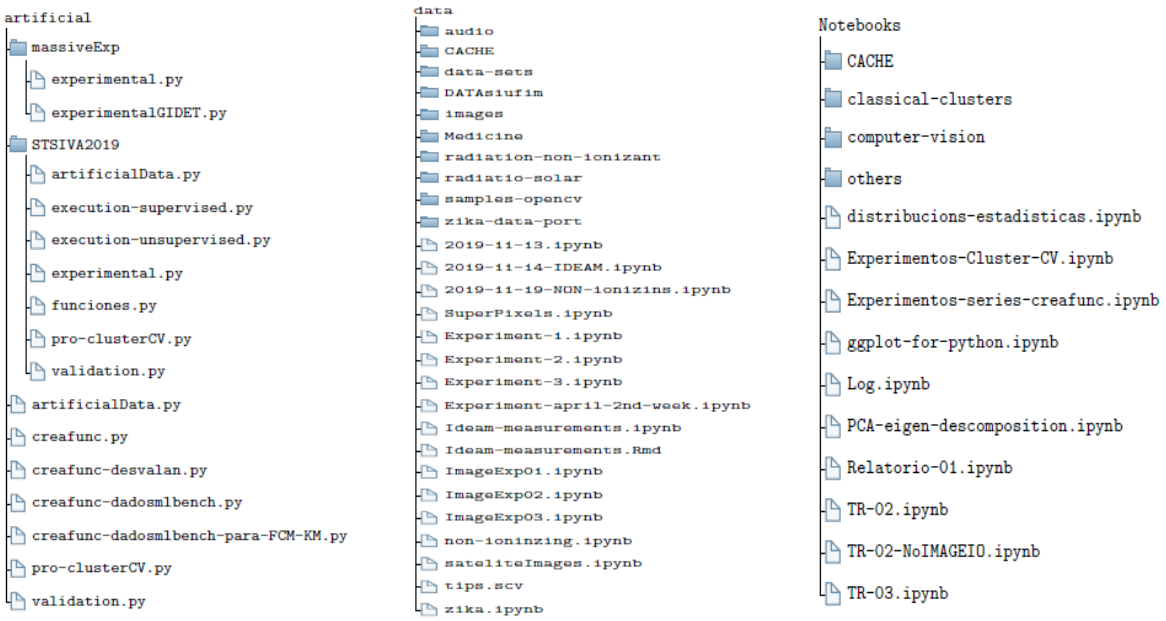
Donde el texto en verde representa funcionalidades y especificaciones del trecho de código siguiente, adicionalmente se añaden comentarios, la mayor parte de ellos se comportan como alternativas secundarias de ejecución o como sugerencias en caso de que algo falle. Por otro lado, el código está estructurado con el paradigma de programación procedimental, la cual permite aplicar la arquitectura modular que presenta el Cluster-CV2, mencionada en la siguiente sección, de esta manera cada código generalizado está hecho de la manera que se lea directamente en lenguaje natural, para el caso de que el usuario lea en inglés. Finalmente, para la documentación del Cluster-CV2 se ha aprovechado la herramienta proporcionada por el GitHub: el "README.md", la cual permite presentar una introducción del aplicativo, añadiendo además de parámetros técnicos, licencia, ejemplos y los autores del trabajo, con sus respectivos currículums vitae (Ver Anexo 1).

3.5.3. Distribución al público.

Teniendo la documentación preparada, se procede a cargar, organizar y liberar el ClusterCV2 bajo el repositorio de GitHub con los códigos en R y Python, creados usando los softwares de Spyder y RStudio, donde la organización de archivos ha quedado como se presenta en la Figura 53, de manera que se evidencien los archivos básicos, la plantilla diseñada a modo de manual de usuario, los datos usados y con los cuales se han desarrollado las pruebas.

Figura 53. Sistema general de archivos en el repositorio de GitHub.

Sin embargo, la Figura 53 solo da un vistazo general del repositorio, debido a la gran cantidad de archivos, principalmente de código, por el que ha sido necesario dar claridad en la Figura 54(a) se presenta los archivos usados en las rutinas de datos sintéticos artificiales, tanto de creación, procesamiento y valoración con las métricas de la matriz de confusión para agrupamiento de datos ; seguidamente, la Figura 54 (b) presenta las mismas tareas de la Figura 54 (a), con la excepción de que estos están enfocados para datos reales.



(a) Tratamiento de datos sintéticos. (b) Tratamiento de datos reales. (c) Pruebas unitarias.

Figura 54. Sistema de archivos en GiHub.

Por último, se tiene la Figura 54(c), en la cual se ha presentado los cuadernos de python y R que han sido usados en las pruebas unitarias, estás han sido las que determinaron los parámetros de valoración en cada operador, donde a medida que se ejecutaban las pruebas, se iba corrigiendo la densidad con la que se aplicaba cada filtro de visión computacional, inclusive desde la reducción de variabilidad en la matriz de distancias.

4. Resultados

Este capítulo presenta los aportes relevantes obtenidos al aplicar la metodología del Cluster CV2 sobre datos sintéticos y reales, las consideraciones tomadas en cuenta para lograrlos y el resumen de los mismo, generalmente en tablas, donde: la primera sección muestra los resultados con las métricas de validación de Silhouette y presision, además de una estimación del costo computacional en la ejecución del algoritmo; en la segunda sección se presenta, la aplicación del Cluster CV2 sobre datos reales, las consideraciones tomadas en cuenta tanto para la adquisición y tratamiento de dichos datos; y finalmente, en la tercera sección se presenta la divulgación de resultados realizadas, acerca de los resultados obtenidos y la funcionalidad del Cluster CV2

4.1. Datos sintéticos

Con el objeto de validar la metodología se han abordado las métricas de validación (Rendón, y otros, 2011): Precision, Recall, Jaccard y Silhouette, para datos sintéticos (Halkidi, Batistakis, & Vazirgiannis, 2002), pues son estos son los que cuentan con información a priori con las cuales evaluar el rendimiento de las operaciones ejecutadas (Halkidi M. a., 2002).

4.1.1. Pruebas unitarias.

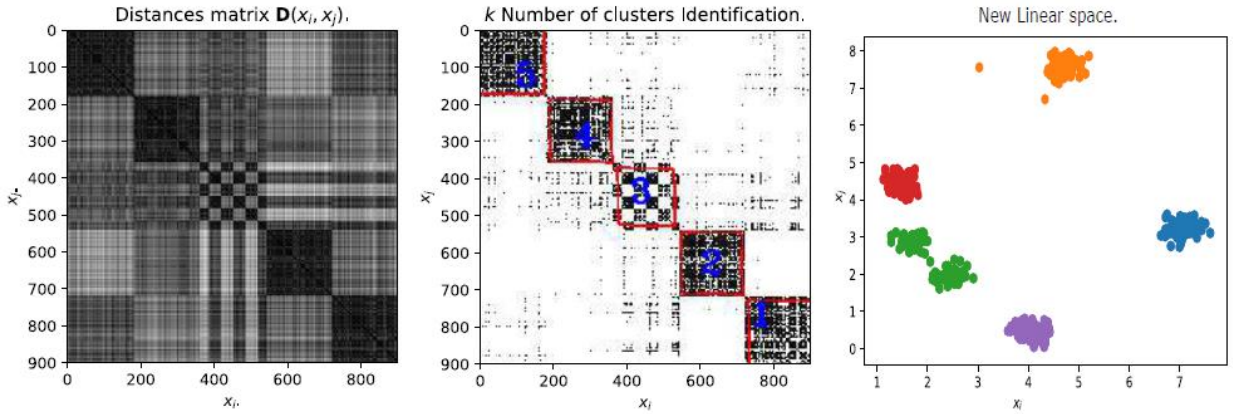
En el diseño de experimentos se había planteado el desarrollo 2 de estos con cada tipo de distribución, donde se iteraba diez veces variando los parámetros del conjunto de datos, es decir, en el experimento uno se había tomado 900 muestras aleatorias usando la distribución Normal, para el cual se varió la desviación estándar desde 0.1 hasta 0.5, siendo este valor, el que representa

el índice de superposición entre muestras, resumiendo los parámetros de experimentación en la Tabla 7. Teniendo en cuenta esto, en la Figura 55 se resume una de las iteraciones realizadas sobre un conjunto de datos sintéticos con 5-grupos definidos previamente, del cual se ha logrado

la correcta identificación usando la matriz de similaridad entre muestras de la Figura 55(a), donde se encuentra la información visual acerca de la afinidad entre muestras, seguidamente al ser procesada con el método de visión computacional de los Algoritmos del 2 al 10, además del algoritmo canny presentado en la Figura 16, cuyos grupos posteriormente fueron sometidos al proceso de descomposición de la matriz de covarianza en auto valores y auto vectores, obteniendo los componentes de dispersión y propagación en los auto vectores, usando el sentido de orientación cartesiano supuesto por las librerías de python y R.

Tabla 7. Modo de experimentación para pruebas unitarias.

Variable	Valores
Distribuciones	Normal, Laplace
Desviación estándar	0,1; 0,2; 0,3; 0,4; 0,5
Número de muestras	900
k-grupos	2; 3; 4; 5
Iteraciones	10



(a) Matriz de similaridad. (b) Identificación espacial. (c) Proyección lineal.

Figura 55. Ejemplo del experimento 1 con 5-grupos, s_d= 0,3 y distribución normal gaussiana.

En la Figura 55 (c) se presenta de esta manera la corrección de la superposición entre muestras al rotar y trasladar los grupos, en función de los ángulos de propagación y dispersión, magnitudes y centroides obtenidos desde el análisis de componentes principales sobre las 2 atribuciones del conjunto de datos, definidas matemáticamente en la ecuación (46), donde sigma es el área de la campana gaussiana tomado, determinado aleatoriamente, los valores de 4 y 2,5 son semillas definidas con el objeto de asegurar un superposición de muestras entre los grupos.

$$x_{in} = \begin{bmatrix} s_d * \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} + 4 \\ s_D * \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} + 2{,}5 \end{bmatrix}^T \tag{46}$$

Finalmente, en la Tabla 8 se presenta las métricas de validación que han evidenciado la concordancia del grupo obtenido para muestra y la precisión en el agrupamiento, donde otros trabajos como (Serra & Tagliaferri, 2016) han obtenido amplias variaciones del $\pm 0{,}1$ entre las métricas de precisión, recall y jaccard, siendo el caso del Cluster-CV2 la variación inferior al $\pm 0{,}00001$, es decir, en el rango de cifras significativas asumidas son iguales, especialmente donde el único trabajo previo que sobrepasó más de 2 cifras significativas ha sido el de (Jaimes, Castro, Torres, Silva, & Braga, 2017).

Tabla 8. Experimento 1 para 5 distribuciones normales con 10 iteraciones.

s_d	Silhouette	Precision	k-grupos
0,1	0,6936±0,0026	0,9951±0,0005	5
0,2	0,5745±0,0095	0,9959±0,0008	5
0,3	0,4470±0,0087	0,9948±0,0019	5
0,4	0,3393±0,0142	0,9928±0,0028	5
0,5	0,2046±0,0155	0,4971±0,0015	4

En este experimento, se presenta un fenómeno similar, pues la identificación de los grupos para el caso de variar la desviación estándar en $s_d = 0{,}5$, significa el unir dos grupos, pues las semillas planteadas aseguran la superposición de muestras de diferentes grupos desde 0,1, tal como se evidencia en la Figura 56, el cual presenta un experimento para la condición mencionada previamente y sus respectivos rótulos después del proceso de identificación por visión computacional.

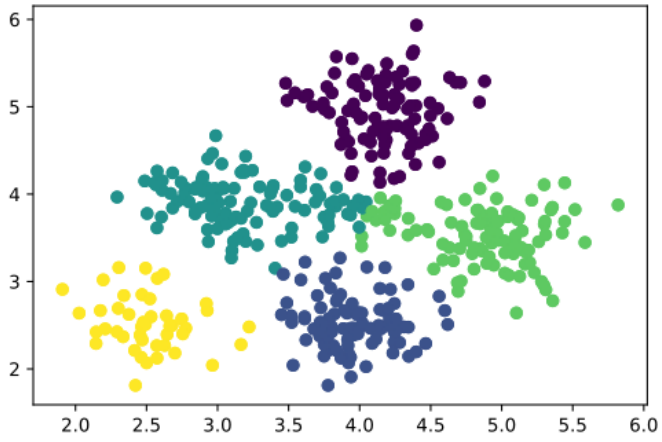


Figura 56. Ejemplo del experimento 1 con 5-grupos, s_d = 0,1 y distribución normal gaussiana.

Continuando, con el experimento número 2, se aborda el caso de muestras desbalanceados, lo cual contrario al anterior experimento, grupo presenta mucho mayor cantidad de muestras con respecto al otro, es decir, en el ejemplo presentado en la Figura 57 se aborda las secuencias de (a)-(c) de la misma manera que se presentó previamente en la Figura 55, siendo para este caso la desviación estándar de $s_d = 0.4$, el número de muestras se mantiene en 900, pero con solo 2-grupos, el cual uno contiene 800 muestras y el otro tan solo 100 muestras.

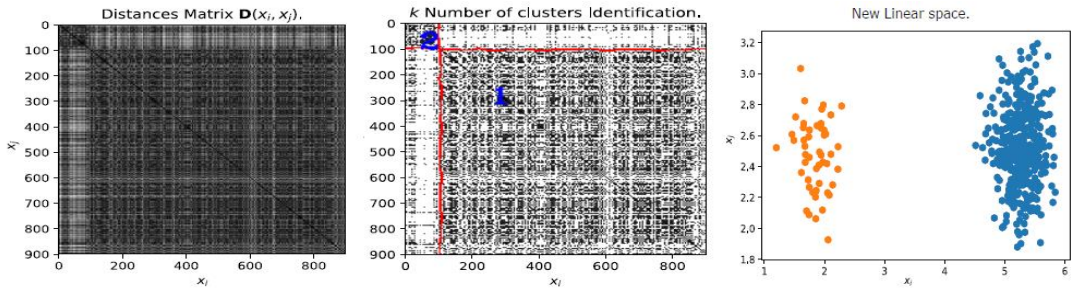


Figura 57. Ejemplo del experimento 2 con 2-grupos, s_d= 0,4 y distribución Laplace.

El comportamiento de la matriz de similaridad se ha mantenido, pero en las operaciones de visión por computadora las operaciones han evidenciado que entre más grande es el nivel de superposición, el grupo con mayor cantidad de muestras tiende extenderse a los alrededores del otro grupo, siendo este el caso extremo, por lo que la condición del centroide ha asegurado la superposición entre las muestras de los grupos, incluso para el nivel más bajo de superposición de $s_d = 0.1$, y evidenciando la conservación de la distribución tanto en R como python (ver la Figura 58). Adicionalmente en la validación, se ha añadido la métrica de Silhouette a la nueva base de datos, obtenida después de aplicar la corrección de la superposición entre muestras, donde los resultados de la Tabla 9 demuestran alta concordancia, aunque la distribución de los datos de la Figura 58 intenta tener una dispersión y propagación alta con respecto al resultado de la Figura 56.

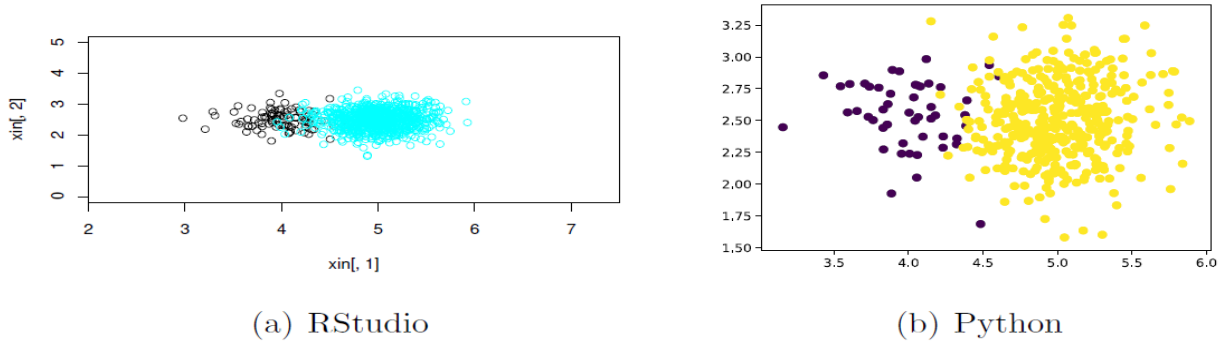


Figura 58. Datos desbalanceados graficados con sus rótulos asignados.

Finalmente, con las ejecuciones de las pruebas unitarias, se ha logrado corregir los parámetros de ventana operativa sobre las matrices de similaridad con las distancias abordadas, siendo para este caso suficiente con la distancia Euclidiana, debido a su comportamiento estadístico, incluso el método de identificación usando visión por computadora adquirió los siguientes kernels funcionales presentados en la Figura 59.

Tabla 9. Experimento número 2 para 2-grupos con datos desbalanceados y 10 iteraciones.

s_d	Silhouette	Precision	Post-silhouette
0,1	0,7989±0,0052	0,9933±0,0000	0,9762±0,0002
0,2	0,6272±0,0037	0,9932±0,0010	0,9568±0,0009
0,3	0,4697±0,0002	0,9946±0,0003	0,9362±0,0013
0,4	0,3664±0,0148	0,9946±0,0054	0,9184±0,0033

```
''' Using : .....'''
criteriaCVopen3x3 = 0.003 # 3x3/9
kernel_3x3 = int(criteriaCVopen3x3*long)
#Another criteria
criteriaCVopen3x3 = 0.003 # 3x3/9
kernel_3x3 = int(criteriaCVopen3x3*long)
criteriaCVopen5x5 = 0.005 # 5x5/25
kernel_5x5 = int(criteriaCVopen5x5*long)
criteriaCVopen20x20 = 0.005 # 20x20/400
kernel_20x20 = int(criteriaCVopen20x20*long)
criteriaCVopen25x25 = 0.03 # 5x5/25
kernel_25x25 = int(criteriaCVopen25x25*long)
```

Figura 59. Kernels funcionales.

4.1.2. Optimización para datos masivos.

Con el algoritmo estable en la identificación, se procede a realizar experimentación masiva, es decir, elevar el número de iteraciones de la Tabla 7 a 100 iteraciones, donde en cada sesión de experimentación usando la metodología de pruebas unitarias se alcanzaba a analizar $3{,}6x10^5$

muestras en aproximadamente 18 horas. Al aumentar el número de iteraciones a 100, ejecutando en arquitectura modular, se alcanza a abordar $3{,}6x10^6$ muestras en un tiempo promedio de 6,6 horas por sesión, consumiendo 880MB de memoria RAM para equipos un equipo con 2.5GHz de procesador y dos núcleos.

Así, se ha obtenido los resultados de la Tabla 10, y al compararlos con la Tabla 8 se encuentra que las métricas reducen su valoración en (0.030 ± 0.001), para el caso del primer experimento.

Tabla 10. Experimento número 1 para 5 distribuciones normales y 100 iteraciones.

s_d	Silhouette	Precision	k-grupos
0,1	0,6904±0,0036	0,9933±0,0000	5
0,2	0,5705±0,0075	0,9933±0,0008	5
0,3	0,4477±0,0132	0,9914±0,0021	5
0,4	0,3091±0,0225	0,9662±0,0584	5
0,5	0,2046±0,0155	0,4971±0,0015	4

Para el caso del experimento número 2, se han obtenido las variaciones, medianamente similares a la Tabla 10, donde la métrica de post-silhoutte sigue arrojando resultados increíbles de concordancia y validez de los grupos obtenidos, tal como se resume en la Tabla 11.

4.1.3. Comparación de resultados.

Como parte de la discusión de los resultados para datos sintéticos, ha sido implementado la técnica k-mean y Fuzzy c-means usando los trabajos de (Zhang, Wang, & Zhang, 2007), aplicando la experimentación masiva sobre las mismas muestras con los dos algoritmos,

obteniendo los resultados de la Tabla 12, en la cual se ha resaltado en negrita aquellos resultados con el mayor índice respecto a la métrica aplicada.

Tabla 11. Experimento número 2 para 2-grupos con datos desbalanceado y 100 iteraciones.

s_d	Silhouette	Precision	Post-silhouette
0,1	0,7992±0.0059	0,9933±0,0000	0.9647±0m0011
0,2	0,6197±0.0138	0,9930±0,0011	0.9315±0,0028
0,3	0,4662±0.0239	0,9932±0,0052	0.8997±0,0067
0,4	0,3551±0.0213	0,9917±0,0035	0.8689±0,0086

De lo cual, el Cluster CV2 ha logrado obtener valoraciones comparables con los algoritmos tradicionales, donde se ha elevado el costo computacional a $O(n^2)$ con respecto al k-means, siendo n el número de muestras contenidas y manteniendo un costo inferior al agrupamiento Espectral $O(n^3)$.

Tabla 12. Comparación de resultados con las técnicas tradicionales.

s_d	Cluster CV2		*k*-means		c-means	
	Silhouette	Precision	Silhouette	Precision	Silhouette	Precicion
0,1	0,6904	0,9933	0,5912	0,5541	**0,8446**	**1,0000**
0,2	0,5705	**0,9933**	0,6894	0,9867	**0,6893**	0,9845
0,3	0,4477	**0,9914**	0,5537	0,4498	**0,5518**	0,4499
0,4	0,3091	**0,9662**	0,4923	0,3734	**0,4909**	0,3722

4.2. Datos reales

Esta sección se ha dividido en los tres análisis completados al momento de la elaboración de este documento, donde primero se aborda la implementación del sistema de adquisición de datos con el objeto de optimizar el proceso de consumo energético en un sistema de incubación de aves domésticas y seguidamente se presenta el abordaje sobre las medidas de radiación no ionizante.

4.2.1. Temperatura, humedad relativa, voltaje y corriente.

Como se mencionó en la metodología, se ha implementado un sistema de adquisición de datos sobre un sistema de incubación de aves domésticas, donde las especificaciones del mismo se deja en el Anexo 3, en esta parte se presenta el análisis de agrupamientos para sistemas termodinámicos y de optimización energética consumida, donde los datos obtenidos consisten en 25 archivos en formato CSV, tomados en 50 días de pruebas y retroalimentación, pues gran cantidad de los datos requirieron ser depurados. Finalmente, dentro del análisis se encontró un gran cantidad de muestras al concatenar todos los archivos, por lo cual se seleccionaron los datos del último experimento con ejecución continua durante 5 horas, arrojando un total de 16429 muestras y 7 atribuciones para cada una, las cuales son: tiempo, temperatura en la fuente de calor, temperatura en la incubadora, humedad relativa en la fuente, humedad relativa en la incubadora, voltaje de la resistencia (fuente de calor) y corriente en la resistencia. Teniendo claro el conjunto de datos, se le aplicado el Algoritmo 2, arrojando la matriz de similaridad de la Figura 60, la cual es la matriz de similaridad entre las muestras del conjunto de datos abordado.

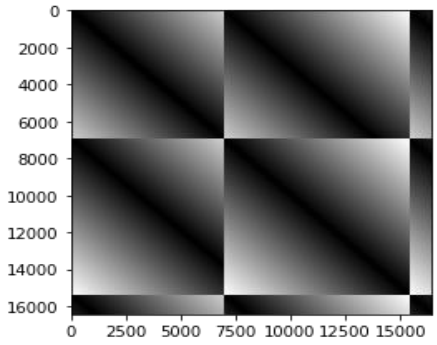
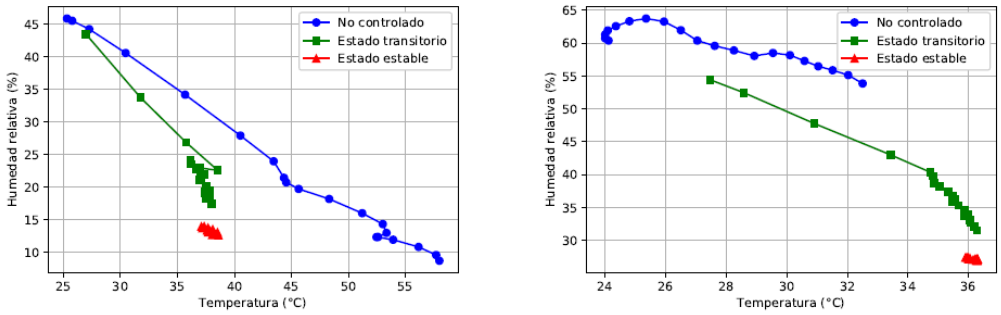


Figura 60. Matriz de similaridad para las 16429 muestras y sus 7 atribuciones.

Debido a que son 16429 muestras, la matriz de similaridad presenta dimensiones 16429×16429, lo cual a nivel computacional tiende a ser costoso, abordar más muestras elevarían significativamente el costo computacional y probablemente se requiera una máquina con más de 4GB de memoria RAM. Sin embargo, con está matriz de similaridad ha sido suficiente para encontrar tres grupos o zonas de operación de un sistema térmico, pues de la Figura 60 se ha obtenido la Figura 61 al aplicar un rotulado clasificar con los grupos obtenidos previamente, en este caso 3.



(a) Fuente de calor. (b) Zona de incubación o carga.

Figura 61. Comportamiento de las variables de temperatura y humedad relativa.

De esta manera, los grupos obtenidos en el trabajo han representado la zona no controlada del sistema, esta zona se presenta en la Figura 62, donde las leyendas han indicado una relación

inversa entre la humedad relativa y temperatura, dependiendo de la zona de afectación en que se está midiendo las variables.

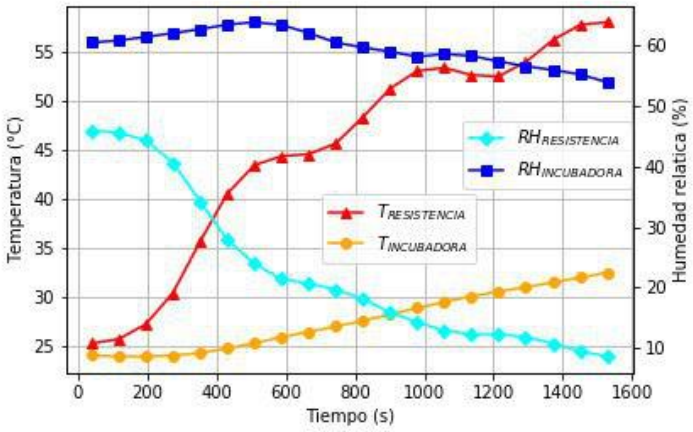


Figura 62. Zona inestable o no controlada del sistema de incubación.

El segundo grupo obtenido ha sido corroborado con las medidas de corriente, pues esta zona pertenece a un estado transitorio de estabilización al actuar controlador binario, el cual asume parámetros de inercia y acumulación de energía, además de la histéresis de los sensores de medición pues, aunque el DHT22 presenta una resolución alta para este proceso, las decisiones basadas en el tiempo y gradiente predominan para alcanzar la estabilidad. Los resultados de estos procedimientos son presentados en la Figura 63 con sus respectivas leyendas.

Finalmente, el último grupo de la Figura 61 representa aquella zona estable, donde las variaciones de las variables de control se han consolidados en rangos pequeños tendiendo a reducir la zona de oscilación como se presenta en la Figura 64, además cabe destacar que a medida que un controlador interviene en la planta, se van reduciendo los rangos de propagación de los datos a través de la escala, siendo el primer grupo el de mayor rango.

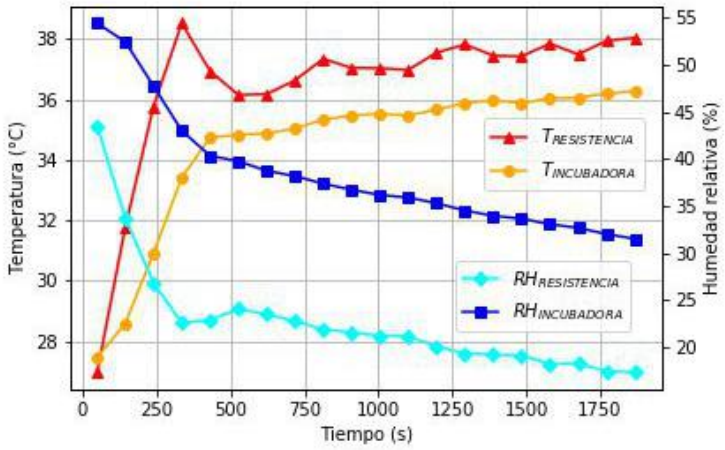


Figura 63. Zona de transición a estado estable.

En este conjunto de datos no ha sido necesario aplicar la corrección de la superposición, pues este conjunto de datos no presentaba superposición.

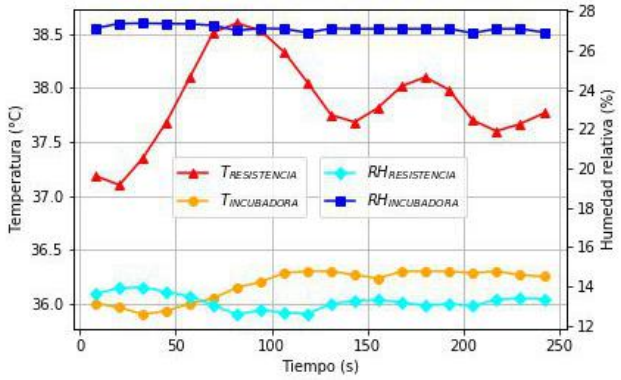


Figura 64. Zona estable de operación del controlador sobre el sistema de incubación.

4.2.2. Medidas de radiación no ionizante.

Continuando con la siguiente base de datos real, se encontrado que la variación de las escalas representa un factor importante en la hora de dar un peso cada una de las características y de esta manera obtener una matriz de similaridad bien comportada, por lo que en la base de datos de radiación no ionizante inicialmente se han normalizado las escala de las mediciones en la Figura 65, donde el campo eléctrico y magnético concuerdan con algunas variaciones, pues se debe errores propias de la incertidumbre del equipo de medida.

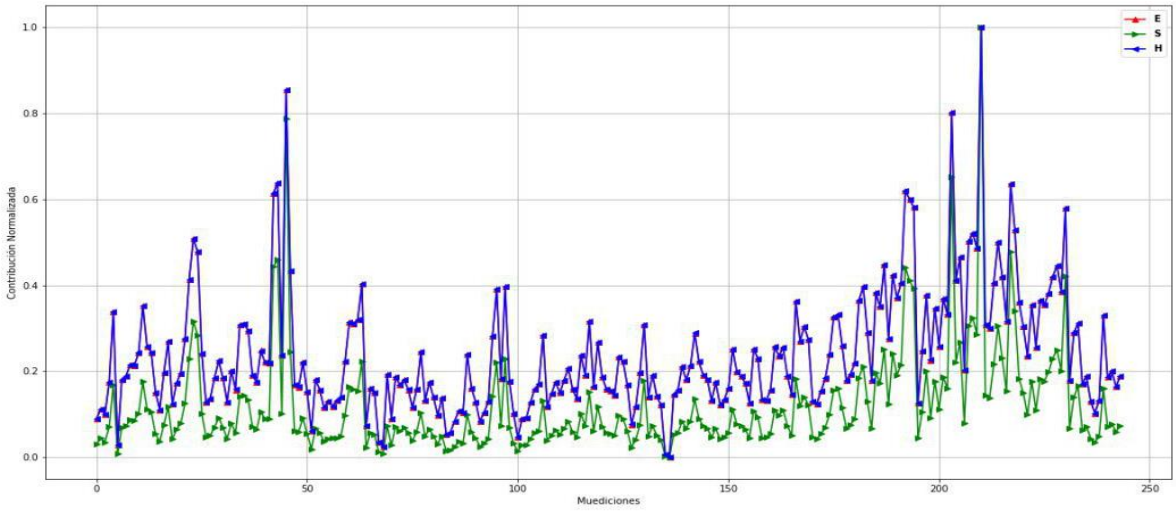


Figura 65. Normalizando las tres variables.

Seguidamente se había planteado la Hipótesis de encontrar los grupos añadiendo atribuciones las tres atribuciones, el campo eléctrico, magnético y potencia de radiación, sin embargo, la matriz de distancias presentada en la Figura 66 ha demostrado que las variables generan una matriz de afinidad con similaridades mal comportadas, sin información relevante acerca de grupos de muestras.

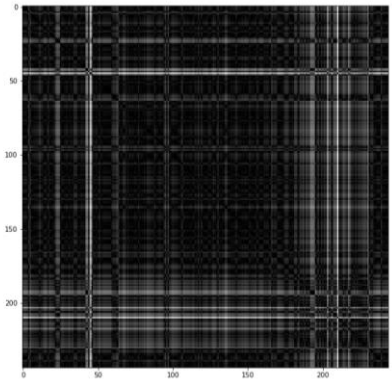


Figura 66. Matriz de similaridad de la Hipótesis número 1.

Por lo que se propuso la supresión del campo magnético de las atribuciones, dejando los campos eléctricos y potencia de radiación, arrojando la matriz de similaridad de la Figura 67, con

un comportamiento similar a la anterior, pues parecía ser que su relación de medio isotrópico no daba información acerca de grupos en el conjunto de datos.

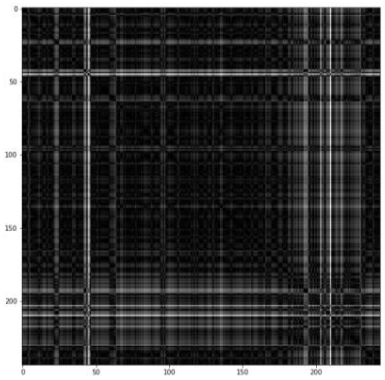
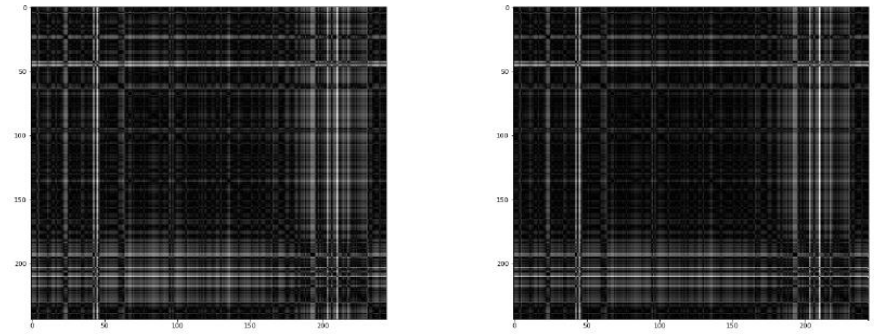


Figura 67. Matriz de similaridad de la hipótesis número 2.

Al ver que esta las dos Hipótesis anteriores no arrojaron resultados que representen los grupos contenidos en el conjunto de datos, se abordó las Hipótesis números 3 y 4 en la Figura 68 siendo la primera con las atribuciones de campos eléctrico y magnético, y la segunda entre el campo magnético y la potencia de radiación, donde se evidencia que la similaridad entre muestras generan una matriz de distancias sin grupos evidentes, inclusive para el método de identificación computacional.

De esta manera la revisión literaria (Jain, Murty, & Flynn, 1999) ha presentado enfoques para aplicar el agrupamiento sobre las características, es decir, aquellos datos donde no se conoce como operar sus atribuciones para conocer los grupos contenidos, se les aplica a los agrupamientos a las atribuciones arrojando la matriz de la Figura 69, con evidentemente 3 grupos en el rango de medición, asumiendo las contribuciones ponderadas en el procedimiento de normalización de las muestras.



(a) Número 3. (b) Número 4.

Figura 68. Matriz de similaridad de las hipótesis.

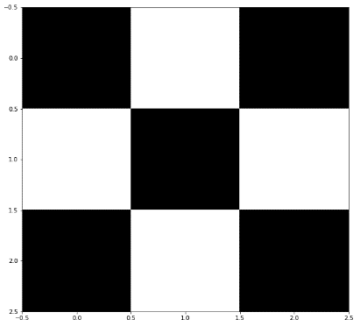


Figura 69. Matriz de similaridad de la hipótesis número 5.

De esta manera se elaboró un análisis estadístico para el trabajo dejado en el Anexo 3, siendo evidentes las tres escalas de exposición a radiación no ionizante.

4.3. Divulgación de resultados

V semana, STSIVA 2019, 5th IMRMPT, 1er Expresa tu ingenio, sustentación pública Finalmente, este trabajo ha sometido los resultados de investigación a pares evaluadores en los cuales se divulga el conocimiento adquirido y brevemente se reaccionarán a continuación:

En el 2019 XXII Symposium on Image, Signal Processing and Artificial Vision (STSIVA), realizado en la ciudad de Bucaramanga, se ha presentado un artículo científico, el cual se encuentra publicado en y dejado en el Anexo 2 del presente documento, con los resultados

obtenidos en la experimentación masiva de datos sintéticos, además de la divulgación de los resultados en el evento en modalidad de ponencia oral (ver Anexo 6). Los resultados para experimentación de pruebas unitarias con datos sintéticos han sido en una ponencia en modalidad de póster para el evento 5th International Week of Science, Technology & Innovation, en el cual se inició la iteración de corrección de parámetros (ver anexo 5).

Por otro lado, el desarrollo del sistema de adquisición de datos y el análisis de ellos han sido divulgado en ponencia de modalidad póster 5th International Meeting for Researchers in Materials and Plasma Technology (ver anexo 7), además de la publicación de un libro de resúmenes con ISSN 2422-3824, y un artículo en la Journal of physics: conference series (ver anexo 3).El sistema de adquisición de datos también ha sido aplicado para la creación de una librería de Arduino en el monitoreo de variables de calidad del aire publicada por Arduino y disponible con identificador único digital (Califa Urquiza, Contreras Contreras, & Ramírez, 2019).

Finalmente, los resultados de radiación no ionizante han sido añadidos al trabajo escrito del Anexo 4 para una presentación en modalidad póster de la Sociedad Colombiana de Ingenieros titulado: I Muestra Nacional de Proyectos en Ingeniería (ver anexo 8).

5. Conclusiones

Los softwares de Spyder y RStudio tienen el paquete de soporte suficiente para desarrollar aplicaciones de agrupamiento de datos, visión por computadora y aprendizaje de máquina, debido a que varios de los algoritmos diseñados, se encontraban previamente en las librerías de estos softwares, incluso con un nivel de optimización significativamente alto.

Los lenguajes de Python y R presentan una sintaxis de alto nivel, especial para desarrollo rápido de aplicaciones, reservando las características de rendimiento y velocidad mínima, logrando los requerimientos funcionales rápidamente, lo cual a través de iteraciones se perfecciona el sistema de agrupamiento y corrección de la superposición.

Los resultados obtenidos al operar datos sintéticos de manera masiva, aproximadamente 3,6 millones de muestras por sesión, han evidenciado la efectividad para cada parámetro entre los que se tiene: el limiar, la ventana de convolución y el factor correctivo, estos han sido los principales potenciadores del algoritmo.

De los anteriores parámetros se ha encontrado que para datos balanceados fue suficiente un valor de umbral en $0{,}9s_d$ en la definición del limiar, representando una variabilidad de los datos inferior al 10%, por otro lado, para datos de desbalanceados se ha determinado que este varía entre $1{,}3s_d$ a $1{,}75s_d$ dependiendo de la relación de un grupo con respecto al otro, pues de ser demasiado pequeño $<<0.001$ no logrará tomarlo en cuenta en la identificación espacial.

La metodología aplicada ha demostrado abordar un problema de análisis de datos con las herramientas de procesamiento digital de señales, aprendizaje de máquina e inteligencia artificial, pues en la parte de pruebas unitarias se establecieron los parámetros sobre los cuales el algoritmo debía operar el conjunto de datos dependiendo de su variabilidad y dimensiones.

La experimentación con los datos de temperatura y humedad relativa han determinado el comportamiento de un sistema de incubación, donde se evidencia como un controlador cambia modifica la propagación de las medidas y establecer un criterio de control más robusto ajustado a las condiciones que alcanza una planta, en el caso abordado, se evidencian 3 zonas las cuales fueron caracterizadas previamente usando la matriz de afinidad.

Por otro lado, el conjunto de datos de radiación no ionizante ha evidenciado la Hipótesis de realizar el agrupamiento sobre las atribuciones del conjunto de datos, de esta manera identificar los rangos de concesión entre las atribuciones y cuales sería sus factores, pues un análisis de cuartiles requiere definir el parámetro de divisiones de la escala en 4, para el que un sistema no paramétrico como el Cluster CV2, lograría dar un vistazo de cómo abordar los datos.

Las métricas de validación han requerido de experimentación masiva, pues al aplicar la metodología de pruebas unitarias, el número de muestras aleatorias abordadas es 10 veces inferior al del sistema de ejecución para datos masivos, consumiendo más tiempo de operación, donde la variación de la métrica se conserva justo después de 100 iteraciones.

Al ejecutar los experimentos 1 y 2 con un algoritmo difuso de c-means usando el k-means, la métrica de silhouette es aproximadamente $0{,}1$ mayor que las obtenidas por Cluster CV2, pero la métrica de precisión disminuye hasta 0.69 cuando el nivel de superposición llega a $s_d = 0.2$ y por $s_d = 0.4$ la precisión alcanza 0.49, es decir, Cluster CV2 ha logrado puntajes superiores de precisión con respecto a las dos técnicas clásicas más usadas, cuando se someten a una situación de datos superpuestos, sin mencionar que estos algoritmos requieren conocer el número *k*-grupos a obtener.

La corrección de la superposición entre muestras ha elevado el nivel de concordancia obtenido con la métrica silhouette, pues al eliminar la superposición entre grupos, las muestras

tienen asignación evidente a un grupo, suprimiendo la incerteza en la superposición de muestras de un mismo grupo.

Otro factor determinante en la corrección de la superposición ha sido el factor multiplicativo, pues este debía ajustarse automáticamente al aplicar el puntaje Fisher, de lo cual se han obtenido variaciones desde 1,4 hasta 1,9.

El tratamiento de información o datos reales generalmente es visto de manera ilegal, aun cuando los datos son públicos, sin la debida autorización de los datos por parte de sus recopiladores, estos no permitirán el procesamiento de la información, por lo que fue necesario desarrollar e implementar el sistema de adquisición de datos, con el objeto de abordar datos reales y exploratorios.

La matriz de distancias brinda información visual en forma general acerca de la relación de similitud del conjunto de datos o conjunto de atribuciones, específicamente los grupos contenidos en el conjunto de datos sin que el especialista en datos esté supervisando cuáles son los grupos k correctos, como se ha visto en los datos de variables termodinámicas y radiación no ionizante.

Recomendaciones

Uno de los principales factores de error este trabajo ha sido el encontrar los bordes óptimamente sobre los grupos, pues esta información no ha sido certera, lo cual requirió desarrollar el cálculo de proximidad al centroide más cercano

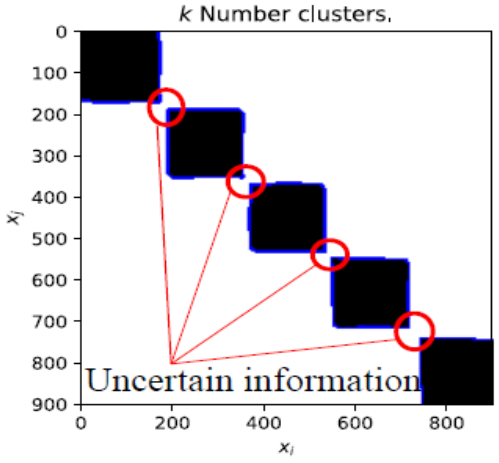


Figura 70. Factores de error.

Se sugiere abordar la metodología sobre datos reales en telecomunicaciones, con el objeto de determinar los factores de impacto en un escenario outdoor o indoor, usando como atribuciones los ángulos azimuth y elevación, además de la métrica Power Delay Profile, pues este tipo de trabajos han sido ampliamente discutidos en la literatura de agrupamiento de datos.

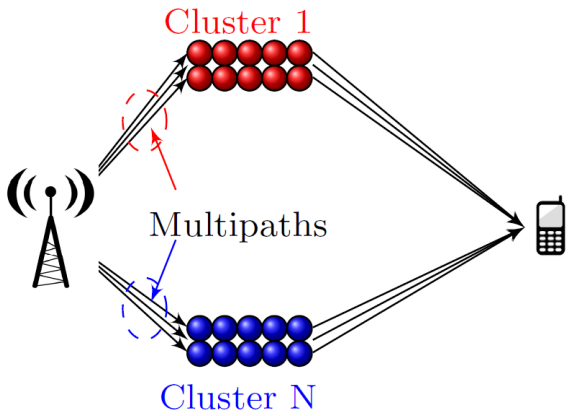


Figura 71. Propuesta de agrupamiento en telecomunicaciones.

Referencias bibliográficas

Alban, N., Laurent, B., Mitherand, N., Ousman, B., Martin, N., & Etienne, M. (2018). Robust and Fast Segmentation Based on Fuzzy clustering combined with Unsupervised Histogram analysis. *IEEE Intelligent Systems*, 1--1.

Beltrán Ortega, P. A., Guevara Ibarra, D., & Mendoza García, J. L. (2018). *Visualización de los niveles de radiación no ionizante en un mapa digital de la ciudad de san José de Cúcuta.* Cúcuta: Universidad Francisco de Paula Santander.

Bi, H., Sun, J., & Xu, Z. (2017). Unsupervised PolSAR image classification using discriminative clustering. *IEEE Transactions on Geoscience and Remote Sensing, 55*(6), 3531--3544.

Bressert, E. (2012). *SciPy and NumPy: an overview for developers.* Bressert, Eli.

Califa Urquiza, M. A., Contreras Contreras, G. F., & Ramírez, J. (Septiembre de 2019). miguel5612/MQSensorsLib: Release library . (1.0.4, Ed.) Cúcuta: Zenodo. doi:10.5281/zenodo.3464092

Celik, T. (2009). Unsupervised change detection in satellite images using principal component analysis and k-means clustering. *IEEE Geoscience and Remote Sensing Letters, 6*(4), 772--776.

Chaves, M. A. (2005). La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software. *InterSedes, 6*(10).

Contreras Contreras, G. F., Dulcé-Moreno, H. J., & Ardila Melo, R. (2019). Arduino data-logger and artificial neural network to data analysis. *Journal of Physics: Conference Series, 1386*, 012070.

Contreras Contreras, G. F., Medina Delgado, B., Guevara Ibarra, D., Leite de Castro, C., & Acevedo Jaimes, B. R. (2019). Cluster CV2: a Computer Vision Approach to Spatial

Identification of Data Clusters. *2019 XXII Symposium on Image, Signal Processing and Artificial Vision (STSIVA).* Bucaramanga.

Davies, E. R. (2012). *Computer and machine vision: theory, algorithms, practicalities.* Academic Press.

De Maesschalck, R., Jouan-Rimbaud, D., & Massart, D. L. (2000). The mahalanobis distance. *Chemometrics and intelligent laboratory systems, 50*(1), 1--18.

Dheeru, D., & Karra Taniskidou, E. (2017). *UCI Machine Learning Repository*. (University of California, Irvine, School of Information and Computer Sciences) Recuperado el 1 de Noviembre de 2019, de archive.ics.uci.edu/ml

Duda, R. O., Hart, P. E., & Stork, D. G. (2012). *Pattern classification.* John Wiley & Sons.

Galluccio, L., Michel, O., & Comon, P. (2005). Unsupervised clustering on multi-component datasets: Applications on images and astrophysics dataUnsupervised clustering on multi-component datasets: Applications on images and astrophysics data. *Signal Processing Conference, 2008 16th European.*

Grabner, M., Bregar, Z., Ivanjko, S., & Valencic, L. (2017). Improved model for the spatial load forecasting of the Slovenian distribution network. *CIRED-Open Access Proceedings Journal, 2017*(1), 2354--2357.

Halkidi, M. a. (2002). Clustering validity checking methods: part II. *ACM Sigmod Record, 31*(3), 19--27.

Halkidi, M., Batistakis, Y., & Vazirgiannis, M. (2002). Cluster validity methods: part I. *ACM Sigmod Record, 31*(2), 40--45.

Hennig, C., Meila, M., Murtagh, F., & Rocci, R. (2015). *Handbook of cluster analysis.* CRC Press.

Hernández Sampieri, R., Fernández Collado, C., & Baptista Lucio, P. (2003). *Metodología de la Investigación.* McGraw Hill Interamericana.

Jaimes, B. A., Castro, C. L., Torres, L. B., Silva, G. L., & Braga, A. P. (2017). Cluster-cv: Uma abordagem de visão computacional para a identificação espacial de agrupamentos de dados.

Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data clustering: a review. *ACM computing surveys (CSUR), 31*(3), 264--323.

Kavitha, K., Sandeep, S., & Praveen, P. (2016). Improved spectral clustering using PCA based similarity measure on different Laplacian graphs. *2016 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC).*

Khanmohammadi, S., Adibeig, N., & Shanehbandy, S. (2017). An improved overlapping k-means clustering method for medical applications. *Expert Systems with Applications, 67*, 12--18.

Kwon, B. C., Eysenbach, B., Verma, J., Ng, K., De Filippi, C., Stewart, W. F., & Perer, A. (2018). Clustervision: Visual supervision of unsupervised clustering. *IEEE transactions on visualization and computer graphics*, *24*(1), 142--151.

Lizárraga, B. (2008). *Agrupamiento de Datos utilizando técnicas MAM-SOM.*

Mantilla, M. C., Ariza, L. L., & Delgado, B. M. (2014). Metodología para el desarrollo de aplicaciones móviles. *Tecnura: Tecnología y Cultura Afirmando el Conocimiento, 18*(40), 20--35.

Mitra, R., & Bhatia, V. (2017). Unsupervised Multistage-Clustering-Based Hammerstein Postdistortion for VLC. *IEEE Photonics Journal*, *9*(1), 1--10.

Molina, J. C., & Moreno, M. E. (2010). Análisis de requerimientos usando BPMN. *Revista Colombiana de Computación*, 85--97.

Mwangi, B., Soares, J. C., & Hasan, K. M. (2014). Visualization and unsupervised predictive clustering of high-dimensional multimodal neuroimaging data. *Journal of neuroscience methods, 236*, 19--25.

Navidi, W. (2006). *Estadística para ingenieros.* McGraw Hill Interamericana.

Ng, A. Y., Jordan, M. I., & Weiss, Y. (2002). On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems.*

Ortiz Diaz, H. A., & Sepúlveda Mora, S. B. (2018). *Estimación de la radiación solar a través de redes neuronales artificiales en la ciudad de Cúcuta.* Cúcuta: Universidad Francisco de Paula Santander.

Parker, J. R. (2010). *Algorithms for image processing and computer vision.* John Wiley & Sons.

Pérez-Suárez, A., Martínez-Trinidad, J. F., Carrasco-Ochoa, J. A., & Medina-Pagola, J. E. (2013). OClustR: A new graph-based algorithm for overlapping clustering. *Neurocomputing, 121*, 234--247.

Perlibakas, V. (2004). Distance measures for PCA-based face recognition. *Pattern recognition letters,* 711--724.

Python language reference, version 3.7. (n.d.). *Python language reference, version 3.7*. (Python Software Foundation Wilmington, DE) Retrieved Noviembre 1, 2019, from python.org

Rath, S. P. (2017). Scalable algorithms for unsupervised clustering of acoustic data for speech recognition. *speechRecognition, 46*, 233--248.

Rendón, E., Abundez, I. M., Gutierrez, C., Zagal, S. D., Arizmendi, A., Quiroz, E. M., & Arzate, H. E. (2011). A Comparison of Internal and External Cluster Validation Indexes.

Proceedings of the 2011 American Conference on Applied Mathematics and the 5th WSEAS International Conference on Computer Engineering and Applications. Puerto Morelos.

Saxena, A., Prasad, M., Gupta, A., Bharill, N., Patel, O. P., Tiwari, A., . . . Lin, C.-T. (2017). A review of clustering techniques and developments. *Neurocomputing, 267*, 664--681.

Serra, A., & Tagliaferri, R. (2016). Unsupervised Learning: Clustering.

Silva, G. R., De Medeiros, R. R., Jaimes, B. R., Takahashi, C. C., Vieira, D. A., & Braga, A. D. (2017). CUDA-Based Parallelization of Power Iteration Clustering for Large Datasets. *IEEE Access, 5*, 27263--27271.

Srinivas, B., & Rao, G. S. (2018). Unsupervised learning algorithms for MRI brain tumor segmentation. *Signal Processing And Communication Engineering Systems (SPACES), 2018 Conference on.*

Tafsast, A., Hadjili, M. L., Bouakaz, A., & Benoudjit, N. (2017). Unsupervised cluster-based method for segmenting biological tumour volume of laryngeal tumours in 18F-FDG-PET images. *IET Image Processing, 11*(6), 389--396.

Tan, K. S., Isa, N. A., & Lim, W. H. (2013). Color image segmentation using adaptive unsupervised clustering approach. *Applied Soft Computing, 13*(4), 2017--2036.

The R Project for Statistical Computing. (2019). *Open Source Initiative*. Retrieved Noviembre 1, 2019, from opensource.org/contact

The R Project for Statistical Computing. (2019). *RStudio*. (The R Foundation) Retrieved Noviembre 1, 2019, from r-project.org/

Tsuda, K., Kawanabe, M., & Müller, K.-R. (2003). Clustering with the Fisher score. *Advances in Neural Information Processing Systems.*

Vora, A., & Raman, S. (2018). Iterative spectral clustering for unsupervised object localization. *Pattern Recognition Letters, 106*, 27--32.

Weinberger, K. Q., & Saul, L. K. (2009). Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 207--244.

Weiss, Y. (1999). Segmentation using eigenvectors: a unifying view. *Computer vision, 1999. The proceedings of the seventh IEEE international conference on.*

Zhang, S., Wang, R.-S., & Zhang, X.-S. (2007). Identification of overlapping community structure in complex networks using fuzzy c-means clustering. *Physica A: Statistical Mechanics and its Applications, 374*(1), 483--490.

Zhao, X., Rangaprakash, D., Denney Jr, T. S., Katz, J. S., Dretsch, M. N., & Deshpande, G. (2018). Identifying neuropsychiatric disorders using unsupervised clustering methods: Data and code. *Data in Brief*.

Zscheischler, J., Mahecha, M. D., & Harmeling, S. (2012). Climate classifications: the value of unsupervised clustering. *Procedia Computer Science, 9*, 897--906.

ANEXOS

Anexo 1. Manual de usuario en GitHub

ClusterCV2

ClusterCV2 is a computer vision approach to idenficate spatially data clusters. This was based on ClusterCV algorithm proposed by Jaimes, Brayan (2017).

Getting Started

```
ProofingGaussianNormal(n_clusters)
```

Prerequisites

You'll need python 3.7.2 or later to run every function included by library. Further, machine requirements must be satisfied using only ARM 1.2GHz and 1GB of ram memory for Linux based distributions, and 2.5GHz dual-core and 4GB or RAM memory for Windows.

Functions:

Function	Inputs	outputs
makeDistancia	Data attribution values	Affinity or similarity matrix
makeNormalizacao	Real samples values	Normalized and modulated samples values
makeLimiar	Normalized samples values and variability thresh	Data samples filtered under maximum variability parameter
makeOrdenacao	Data samples	Hieraro clustering labeling
makeMethodImg	Affinity or similarity filtered matrix	k clusters and labels for every sample
makeOverlapCorre	Data labeled	New data set without overlap issue and projected into a new linear space

Installing

Clone this repositry into your desktop or micro computer.

```
git clone https://github.com/Ghiordy/Cluster-CV2
```

Running the tests

Use learning systems based on massive data and simultaneously saving validation results for each iteration and summarizing it into a confusion matrix plot, silhouette scores, and comparison outputs vs outputs parameters.

Learning system

This evaluate, test and proof a specific data set, generally supñied by user, where he doesn't know nothing about itself.

```
LearningSystem(dataSet).py
```

And coding style tests

These tests may generate statistics validation using descriptive tools for cuantitative variables.

```
testStatics.ino
```

Built With

- ClusterCV (https://github.com/P4yo/ClusterCV) - ClusterCV propose based on computer vision.
- Open CV (https://github.com/opencv/opencv) - Computer vision tools for python.

Contributing

Please read CONTRIBUTING.md (https://github.com/Ghiordy/Cluster-CV2/blob/CONTRIBUTING.md) for details on our code of conduct, and the process for submitting pull requests to us.

Authors

- **Brayan R. Acevedo J.** - *GitHub* (https://github.com/P4yo) - CV (http://buscatextual.cnpq.br/buscatextual/visualizacv.do?id=K8118317T7)
- **Byron Medina Delgado** - CV (http://scienti.colciencias.gov.co:8081/cvlac/visualizador/generarCurriculoCv.do?cod_rh=0000290157)
- **Ghiordy F. Contreras C.** - *GitHub* (https://github.com/Ghiordy) - CV (https://scienti.colciencias.gov.co/cvlac/visualizador/generarCurriculoCv.do?cod_rh=0000050476)

See also the list of contributors (https://github.com/Ghiordy/Cluster-CV2/contributors) who participated in this project.

License

This project is licensed under the MIT License - see the LICENSE.md (LICENSE.md) file for details

Anexo 2. Artículo publicado la IEEE Xplore Digital Library.

2019 XXII Symposium on Image, Signal Processing and Artificial Vision (STSIVA)

Cluster CV2: a Computer Vision Approach to Spatial Identification of Data Clusters

Ghiordy Ferney Contreras Contreras
Electricidad & Electrónica
Universidad Francisco de Paula Santander
Cúcuta, Colombia
ghiordy@ieee.org

Byron Medina Delgado
Electricidad & Electrónica
Universidad Francisco de Paula Santander
Cúcuta, Colombia
byronmedina@ufps.edu.co

Dinael Guevara Ibarra
Electricidad & Electrónica
Universidad Francisco de Paula Santander
Cúcuta, Colombia
dinaelgi@ufps.edu.co

Cristiano Leite de Castro
Electrical Engineering
Univeridade Federal de Minas Gerais
Belo Horizonte, Brazil
crislcastro@gmail.com

Brayan Rene Acevedo Jaimes
Electrical Engineering
Universidade Federal de Minas Gerais
Belo Horizonte, Brazil
payo@ufmg.br

***Abstract*—This work shows a novel application based on techniques of Computer Vision and Machine Learning to identify k clusters into a data set with overlapping issue. Used in area of unsupervised data clustering, where separation between groups is tricky. Through pair-to-pair distance calculations upon original data, is gotten a Distances Matrix as representative information of data. This matrix contains visual information, then using morphological operators extract relevant features for individual identification of groups in data set. Next, matrix decomposition performed to covariance matrix, being calculation of data elements for each cluster in order to project data into a new linear space. So, overlapping and separation distances among clusters are corrected without loss information. Results present correct identification of k clusters, without loss information, and eliminating data overlap. Clustering validation metrics such as Silhouette and Precision was used to test the methodology.**

***Index Terms*—Unsupervised clustering, data overlapping, computer vision, machine learning.**

I. Introduction

Clustering is the process of grouping together similar samples into distinct partitions, being a common type of unsupervised machine learning and can be useful for summarizing and aggregating complex multi-dimensional data to make it more interpretable for analysis [1]–[3], e.g. symptoms suffered by people with the same disease, when other person come in this group, that person currently has suffered the same symptoms referred to whose disease [4], [5].

The most of real issues isn't a path to identify labels for every sample, implicating high computer effort and cost, as well as there is not prior information about groups in data set. On other hand, unsupervised clustering has been used upon data mining with big data content, where data clustering aims to find the appropriate groups with minimum samples spread and maximum clusters separation. Generally, a data set has its samples scattered randomly over linear space, where a sample could stay over other or other samples, limiting visual identification or recognition for groups, this condition is currently known as overlapping issue explained separately in the next section [6], [7]. Clustering may be made using a similarity criterion for a individual sample respect others into data set. Where this criterion will give relevant information about of how a sample relations with other, being applied this procedural for all samples to obtain samples groups and their labels [8]. Dependently of approach, literature has several algorithms for cluster detection, the popular algorithms are: k-means [9]–[12], fuzzy c-means [6], [13], hierarchical clustering [14], spectral clustering [15], [16], and others. Cluster analysis requires similarity metric for identification and being spatial identification, there are several approaches where Euclidean, Minkowski, Manhattan, Hamming, and Mahalanobis are widely used. For this works is taken Euclidean Distance Metric (EDM) for similarity samples relations, then applying morphological operators of computer is obtained groups and labels. Finally classical eigenvalue descomposition over covariance matrix to correct overlapping issue and project over a new linear space.

II. Data overlapping issue

Overlapping data has increades interest of researchers during last years, due to this kind of data distribution trends to generate undesirable clusters [7], where many algorithms in cluster analysis differ in how different groups relate to each other. Many cluster approach manages partitional approach where $C_j \cap C_k = 0$, that is, data samples for different groups haven't relationship [17]. Therefore, hierarchical is a special case used especiallely for data with samples overlapped, being a sequence of partitions $C = \cup_{j=1}^{m} C_j$, but these partitions have relations between them $C_j \cap C_k \neq 0$ (see Fig. 1) allowing relations in samples from different clusters [18].

$$val = \frac{d_{\text{inter-samples}}}{d_{\text{inter-clusters}}} = \frac{\frac{1}{N}\sum_{i=1}^{k}\sum_{x \varepsilon C_i} \parallel x - z_i \parallel^2}{min_{i,j}\left(\parallel z_i - z_j \parallel^2\right)} \quad (1)$$

978-1-7281-1491-0/19/$31.00 ©2019 IEEE

Validate the relation for inter-clusters distances and inter-samples distances is possible with equation (1) defining the correct number of clusters such as number of clusters is k, i is a constant defined by $i = 1, 2, 3, ..., k-1$, and in the same way for constant j as $j = i+1, i+2, ..., k$, data size is N, z_i represent current centroid for each cluster defined as C_i.

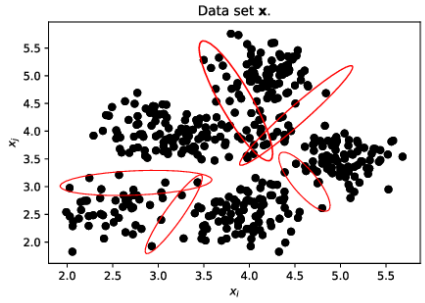


Fig. 1. Data overlapping for synthetic data set with 0.3 standard deviation (s_d). Red ellipses signs data overlapping inter-clusters, where x_i axis is fist instance for data (1-D), and x_j axis represent data for second instance (1-D).

III. Methods and Procedures

This work has been focused to spatial identification of data clusters, where scientific literature has the EDM as the most used metric for similarity between samples into a same data set. The methodology purposed has two stages: the first stage find the correct number of clusters (k-clusters) contained in data set, then split each data group using Computer Vision Techniques upon distances matrix with relevant visual information about of samples relations. Second stage is the linear projection for the k-clusters detected in order to correct overlapping between clusters, that is, increase inter-clusters distance when inter-samples distance be reduced until inter-clusters distance be greater than inter-samples distance.

A. Identification of k number of clusters

Initially, data set is uploaded into a variable named $\mathbf{x}$, this variable is a matrix with a shape of $N \times \ell$, where N is the data size and ℓ are the instances for every sample. Next, must be computed distances matrix pair-to-pair, using EDM as value for similarity degree into samples defined in equation (2) like:

$$D(x_i, x_j) = \sqrt{\sum_{n=1}^{\ell} (x_{i\ell} - x_{j\ell})^2} \tag{2}$$

where ℓ is length for each sample (instances), thus this work that value is initially used with $\ell = 1, 2$ that is into 2 dimensions (2-D). Distances matrix has visual information respect to similarity relationship between pair-to-pair samples, and Fig. 2 shows darkness region as more similarity zones, otherwise lighter zones as less similarity zone for each sample. So, for two samples tested matrix will present high values when its colors differs from black, being black when the similarity is closest, and due to RGB components are increased to obtain light colors when similarity is lesser.

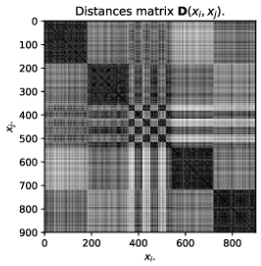


Fig. 2. Distances matrix defined by equation $\mathbf{D}$.

Fig. 2 also presents five blocks or dark regions over principal diagonal in matrix, these regions represent high similarity data into samples for these regions, thus each region must be divided into k-clusters identified and label each sample for all samples involved in data set. For it, computer need recognize like a human using computer vision techniques. Computer vision analysis is a robust manner to find quickly clusters highlighted for distances matrix, but is not enough for correct identification, due to there are several features who can generate wrong identification like noise. Therefore, distances matrix is carried out to apply a threshold into $[0-1]$ scale and computing standard deviation for all distances matrix's values (see Fig. 3). Next, is realized the first threshold using $0.9s_d$ as limit, whose value was defined experimentally on results with synthetic data sets and being expressed in equation (3).

$$L(x_i, x_j) = \begin{cases} 1, & \text{if} \quad M(x_i, x_j) \geq 0.9 s_d \\ M(x_i, x_j), & \text{if} \quad M(x_i, x_j) < 0.9 s_d \end{cases} \tag{3}$$

Threshold matrix ($L(x_i, x_j)$) is saved into a image format (PNG - *Portable Network Graphics*) to go to Computer Vision analysis and operations. Coming, Computer Vision stage starts with blurring filter, this filter is expressed by equation (4)

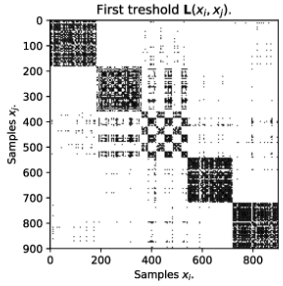


Fig. 3. Distances matrix after first threshold $\mathbf{L}$.

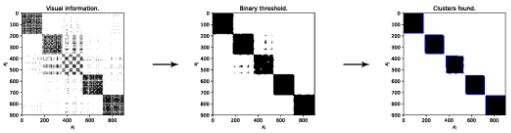


Fig. 4. Computer Vision analysis over distances threshold matrix $L(x_i, x_j)$.

scoping to smoothed image reducing texture and homogenize noise into image.

$$G(x_i, x_j) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x_i^2 + x_j^2}{2\sigma^2}\right) \tag{4}$$

Later, image passes for binary threshold filter that facilitate edge detection or regions with clusters, against is smallest noise zone or pixels with disliked values, these pixels are deleted using erode and dilate operators over image with a *kernel* matrix such as size is 20×20 pixels. Finally to find contours let's use Canny and Sobel edge detectors for k-clusters identification [19]. Fig. 4 has the three main involved stages for Computer Vision analysis applied with a data set with 900 samples and two dimensions (2-D).

B. Linear projection into a new linear space

This section aims to a new linear space where data will be projected in order to correct overlapping inter-clusters, maintaining all original information. Equation (1) represent a validation metric, these metric define our procedure to obtain inter-clusters distances greater than inter-samples distances. But spread degree isn't unknown for every cluster into data set, this value determines how much move data toward their centroid, being it an overlapping suppression. Thus, a recursive way to know spread degree is through covariance matrix ($\mathbf{\Sigma}$ in equation (5)) using variance as $\left(\sigma_{x_i}^2 = \frac{1}{N-1}\left[\sum_{i=1}^{N}(x_i - \mu)^2\right]\right)$ and standard deviation ($s_d = \sqrt{\sigma^2}$) terms, where variance term defines spread degree of data and standard deviation fed scattered data measure over 2-D features space [18]. Hence, covariance matrix summaries all information about of data spread degree on round axis x_i and x_j, respectively.

$$\mathbf{\Sigma} = \begin{bmatrix} \sigma(x_i, x_i) & \sigma(x_i, x_j) \\ \sigma(x_j, x_i) & \sigma(x_j, x_j) \end{bmatrix} \tag{5}$$

Mean value μ_{x_i} is defined as $\frac{1}{N}\sum_{n=1}^{N} x_{in}$ and a similar way for μ_{x_j}. Therefore, Algorithm 1 evidence linear projection process into data set. Algorithm 1 iterates k number of clusters detected using Computer Vision analysis from the previous section, for every clusters is obtained information about its spread degree and must be analyzed separately through eigen-vectors and eigen-values.

In algorithm 1 only are required two general procedures to correct data overlapping, when is knew data spread degree, being q the multiplicative factor of separation varied by user, which determines how much the groups will move into from the initial linear space.

Algorithm 1 Linear projection's algorithm

for $g_k \leftarrow 1, \text{until } k$ **do**
  $[\mathbf{D}_k, C_{ik}, C_{jk}] \leftarrow [x_{ik}, x_{jk}, idx_k]$ ▷ Extract data
  $\mathbf{\Sigma_k} \leftarrow [\mathbf{D}_k \mathbf{D}_k^T, C_{jk}, C_{ik}]$
  $(\overrightarrow{v}_k, \lambda_k) \leftarrow \Sigma_k \overrightarrow{v}_k = \lambda_k \overrightarrow{v}_k$ ▷ Covariance matrix
  $S = \begin{bmatrix} s_{x_i} & 0 \\ 0 & s_{x_j} \end{bmatrix}$ $R = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$
  procedure TRANSLATION($\mathbf{D}_k, R, S$)
    $(T_{ik}, T_{jk}) \leftarrow T = RS \rightarrow (\mathbf{D}_k)$
    $\mathbf{D}_k \leftarrow \mathbf{D}_k + T$ ▷ $\downarrow d_{\text{inter-samples}}$
  end procedure
  procedure REPULSION(r_{x_i}, r_{x_j})
    if $r_{x_i} \neq 0$ **then**
      $r_{x_i} \leftarrow -q * r_{x_i}$
    end if
    if $r_{x_j} \neq 0$ **then**
      $r_{x_j} \leftarrow -q * r_{x_j}$
    end if
    $\mathbf{D}_k \leftarrow \mathbf{D}_k + r$ ▷ $\uparrow d_{\text{inter-clusters}}$
  end procedure
end for

IV. EXPERIMENTAL RESULTS AND DISCUSSION

Aiming to evaluate and review clusterCV2 behavior for spatial identification and linear projection over data sets with overlapping issue, was been made two experiments. In the first experiment was used 5 Gaussian normal distributions varying overlap level defined by standard deviation into a range $[0.1 - 0.5]$, where algorithm achieves to detect correctly k-clusters from a data set, except for $s_d = 0.5$. In case when $s_d = 0.5$ the experiment detected correct k-clusters into several proofs, but the most of tests gives 4 clusters into a data set, thus for this standard deviation was evaluated for cases with 4 clusters was detected. Table I show experiment 1 results using the 5 standard deviations such as was evaluate Silhouette metric as concordance grade and Precision metric as accuracy metric [20].

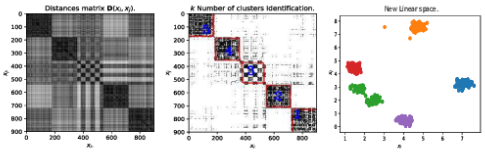
TABLE I
EXPERIMENT 1 FOR 5 GAUSSIAN DISTRIBUTIONS.

s_d	Silhouette	Precision	k
0.1	0.6904 ± 0.0036	0.9933 ± 0.0000	5
0.2	0.5705 ± 0.0075	0.9933 ± 0.0008	5
0.3	0.4477 ± 0.0132	0.9914 ± 0.0021	5
0.4	0.3091 ± 0.0225	0.9662 ± 0.0584	5
0.5	0.2046 ± 0.0155	0.4971 ± 0.0015	4

With Silhouette metric we can deduce how much concordance presents the labels obtained with Computer Vision procedures, where the only one information given to this metric was the labels gotten, data input matrix, and metric used for similarity relations. For other hand, Precision metric gives how much similar are the gotten labels respect to the truth labels, but how to know the truth labels? for it synthetic data was used to test clusterCV2 behavior, supposing that every group is generated with its respective labels, later these

groups are concatenated into a singular matrix and finally data is unsorted randomly to avoid recognition directly from order in samples.

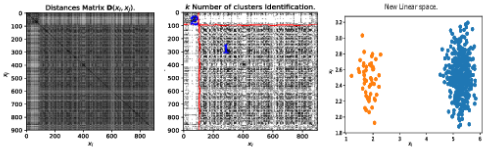
Then, testing 100 times each case of standard deviation we can define average metric value and deduce for $s_d = 0.5$ that precision validation metric is highly affected when isn't detected a correct k-clusters contained into data set. Fig. 5 illustrates a proof for a $s_d = 0.4$ in experiment 1, where (a) is a similarity image as input obtained when euclidean distance was applied pair-to-pair over all samples contained into data set, (b) is k-clusters identification based on computer vision morphological operations, and (c) is linear projection into a new linear space through principal component of analysis and covariance tools, eliminating data overlapping as well as inter-cluster distances was increased when inter-samples distances was reduced.



(a) Similarity matrix. (b) Spatial identification. (c) Linear projection

Fig. 5. Experiment 1.

Continuing with experiment 2 we've tested using unbalanced groups or data set unbalanced. That is, there are clusters with greater number of samples in a cluster respect of them. So thus, it's generated a data set with 900 samples using 2 Gaussian normal distributions, one of them will have 100 samples and other 800 samples in order to accomplish unbalanced condition. Fig. 6 shows a similar analysis for unbalanced data as Fig. 5 except now are 2 Gaussian normal distributions, such as (a) is image obtained applying pair-to-pair euclidean distances over all samples, (b) is Computer vision result for spatial identification, and (c) is linear projection into a new linear space and overlapping correction with a multiplicative factor in $q = 1.9$.



(a) Similarity matrix. (b) Spatial identification. (c) Linear projection

Fig. 6. Experiment 2.

For unbalanced data is necessary modify first threshold to 1.5, due to similarity matrix showed in Fig. 6a present greater noise than experiment 1. Also was tested with threshold between 1.5 and 2 presenting good performance and achieving correct identification of k-clusters. In the same experiment the Linear projection was fixed the multiplicative value to $q = 2.4$, thus was obtained data without overlapping issue. Table II show validation measures for standard deviation between 0.1 and 0.4, for standard deviation equals to 0.5 we gotten that there isn't a correct identification of clusters like experiment 1 proofs. Finally we've added to Table II Post-silhouette column correspondent a concordance metric applied after linear projection has been performed, in this column was improved results due to correlation between metric applied and cluster obtained.

TABLE II
EXPERIMENT 2 FOR 2 GAUSSIAN DISTRIBUTIONS WITH DATA UNBALANCED.

s_d	**Silhouette**	**Precision**	**Post-silhouette**
0.1	0.7992 ± 0.0059	0.9933 ± 0.0000	0.9647 ± 0.0011
0.2	0.6197 ± 0.0138	0.9930 ± 0.0011	0.9315 ± 0.0028
0.3	0.4664 ± 0.0239	0.9932 ± 0.0052	0.8997 ± 0.0067
0.4	0.3551 ± 0.0213	0.9917 ± 0.0035	0.8689 ± 0.0086

V. CONCLUSION

The purpose of researching has been achieved into the development for this work through computer vision techniques and principal component of analysis tools. Distances matrix gives visual information in general form about of data set similarity relation, specifically the clusters contained in data set without data specialist is supervising what it is the correct k-clusters. Thus, spatial identification through Computer Vision analysis was robust manner for variations of overlapping level denoted like standard deviation, making good results for unsupervised clustering. On other hand, morphological operators need contours notably defined upon image, raising level analysis for edge detection and recurring to centroids values for attribute each of sample located on border zone. Executing the same experiment 1 with fuzzy c-means algorithm using predefined k-cluster we found that silhouette metric is approximately 0.1 greater than cluterCV2, but precision metric decrease until 0.69 when $s_d = 0.2$ and for $s_d = 0.4$ precision reaches 0.49, that is, clusterCV2 has more precision over data with overlapping issue. Also, classical k-means algorithm was applied over experiment 1 with k-clusters was predefined too, where precision results are significantly less than clusterCV2 and silhouette results are greater only when $s_d \leq 0.3$. Finally, linear projection achieves correct overlapping issue over original data set, that is, distances inter-clusters is greater than distances inter-samples. Future work must improve Computer Vision analysis and adaptive input constants like are multiplicative factor q and first threshold value L.

REFERENCES

[1] B. C. Kwon, B. Eysenbach, J. Verma, K. Ng, C. De Filippi, W. F. Stewart, and A. Perer, "Clustervision: Visual supervision of unsupervised clustering," *IEEE transactions on visualization and computer graphics*, vol. 24, no. 1, pp. 142–151, 2018.

[2] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM computing surveys (CSUR)*, vol. 31, no. 3, pp. 264–323, 1999.

[3] B. Lizárraga, *Agrupamiento de Datos utilizando técnicas MAM-SOM*. PhD thesis, Tesis profe, 2008.

[4] A. Saxena, M. Prasad, A. Gupta, N. Bharill, O. P. Patel, A. Tiwari, M. J. Er, W. Ding, and C.-T. Lin, "A review of clustering techniques and developments," *Neurocomputing*, vol. 267, pp. 664–681, 2017.

[5] S. Khanmohammadi, N. Adibeig, and S. Shanehbandy, "An improved overlapping k-means clustering method for medical applications," *Expert Systems with Applications*, vol. 67, pp. 12–18, 2017.

[6] S. Zhang, R.-S. Wang, and X.-S. Zhang, "Identification of overlapping community structure in complex networks using fuzzy c-means clustering," *Physica A: Statistical Mechanics and its Applications*, vol. 374, no. 1, pp. 483–490, 2007.

[7] B. A. Jaimes, C. L. Castro, L. B. Torres, G. L. Silva, and A. P. Braga, "Cluster-cv: Uma abordagem de visão computacional para a identificação espacial de agrupamentos de dados," 2017.

[8] E. López-Rubio, E. J. Palomo, and F. Ortega-Zamorano, "Unsupervised learning by cluster quality optimization," *Information Sciences*, vol. 436, pp. 31–55, 2018.

[9] B. Mwangi, J. C. Soares, and K. M. Hasan, "Visualization and unsupervised predictive clustering of high-dimensional multimodal neuroimaging data," *Journal of neuroscience methods*, vol. 236, pp. 19–25, 2014.

[10] A. Tafsast, M. L. Hadjili, A. Bouakaz, and N. Benoudjit, "Unsupervised cluster-based method for segmenting biological tumour volume of laryngeal tumours in 18f-fdg-pet images," *IET Image Processing*, vol. 11, no. 6, pp. 389–396, 2017.

[11] B. Srinivas and G. S. Rao, "Unsupervised learning algorithms for mri brain tumor segmentation," in *Signal Processing And Communication Engineering Systems (SPACES), 2018 Conference on*, pp. 181–184, IEEE, 2018.

[12] J. Zscheischler, M. D. Mahecha, and S. Harmeling, "Climate classifications: the value of unsupervised clustering," *Procedia Computer Science*, vol. 9, pp. 897–906, 2012.

[13] N. Alban, B. Laurent, N. Mitherand, B. Ousman, N. Martin, and M. Etienne, "Robust and fast segmentation based on fuzzy clustering combined with unsupervised histogram analysis," *IEEE Intelligent Systems*, pp. 1–1, 2018.

[14] R. Mitra and V. Bhatia, "Unsupervised multistage-clustering-based hammerstein postdistortion for vlc," *IEEE Photonics Journal*, vol. 9, pp. 1–10, Feb 2017.

[15] G. R. L. Silva, R. R. De Medeiros, B. R. A. Jaimes, C. C. Takahashi, D. A. G. Vieira, and A. D. P. Braga, "Cuda-based parallelization of power iteration clustering for large datasets," *IEEE Access*, vol. 5, pp. 27263–27271, 2017.

[16] A. Vora and S. Raman, "Iterative spectral clustering for unsupervised object localization," *Pattern Recognition Letters*, vol. 106, pp. 27–32, 2018.

[17] A. Serra and R. Tagliaferri, "Unsupervised learning: Clustering," 2016.

[18] C. Hennig, M. Meila, F. Murtagh, and R. Rocci, *Handbook of cluster analysis*. CRC Press, 2015.

[19] J. R. Parker, *Algorithms for image processing and computer vision*. John Wiley & Sons, 2010.

[20] M. Halkidi, Y. Batistakis, and M. Vazirgiannis, "Cluster validity methods: part i," *ACM Sigmod Record*, vol. 31, no. 2, pp. 40–45, 2002.

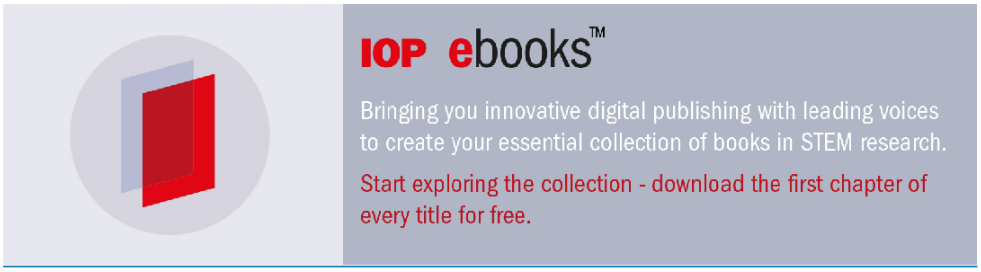
Anexo 3. Artículo publicado en la Journal of Physics: Conference Series

Journal of Physics: Conference Series

PAPER • **OPEN ACCESS**

Arduino data-logger and artificial neural network to data analysis

To cite this article: G F Contreras Contreras *et al* 2019 *J. Phys.: Conf. Ser.* **1386** 012070



5th IMRMPT IOP Publishing
Journal of Physics: Conference Series **1386** (2019) 012070 doi:10.1088/1742-6596/1386/1/012070

Arduino data-logger and artificial neural network to data analysis

G F Contreras Contreras[1], H J Dulcé-Moreno[2,3], and R Ardila Melo[2]

[1] Grupo de Invesitgación y Desarrollo en Electrónica y Telecomunicaciones, Universidad Francisco de Paula Santander, Cúcuta, Colombia
[2] Grupo de Investigación en Física y Materia Condensada, Universidad Francisco de Paula Santander, Cúcuta, Colombia
[3] Materials Science and Technology Research Group, Foundation of Researchers in Science and Technology of Materials, Colombia

E-mail: ghiordyferneycc@ufps.edu.co, hectorjaimedm@ufps.edu.co

Abstract. This work takes thermodynamic modelling through computer science for incubation process at domestic birds, that has presented energy consumption significantly high than energy used in processes. Thus, a data analysis was applied upon variables of temperature and relative humidity for heating zones, trying to know how much energy supplied by source was used, as well as, voltage and current variables are measured in the same moment that temperature and relative humidity are acquired. Then, data analysis was done using artificial neural networks models with samples obtained from sensors, where real process is highly time-variant, fixing environment conditions at the moment required. Therefore, with this system has been obtained an air flow of 3.4375 10^{-2} m^3/J using a anemometer respect to electrical energy supplied by fans, giving 9.4818 W of average power using ceramics resistances, and testing an adaptive controller where its variables are fitted using equations obtained from data analysis. In contrast, colombian farmers have decreased economic conditions to maintain them productions due to free trade agreements implemented lastly, indeed this system was developed using open-source software and hardware to avoid costs in acquisition by licensing politicians or periodic subscription to a specific product developed by companies.

1. Introduction

Thermodynamic systems have a process that include time-variant heating changes dependently from environment coordinate at the moment [1], theory's equations give a general description for physics phenomena, where the most have assumptions or only was designed for a specific number of variables, with static behavior over time [2]. Real process are highly time-variant, for it is known like dynamic process, and its behavior dependent of targeted variables.

Actually exists massive sensors in commercial market to all type of physics variable and counting with easy connection to digital systems like microcontrollers, microprocessors, digital signal processors (DSPs), and others [3, 4]. Through these sensors can obtain data and manage itself using computers with math operations, where data is so important to know how a system changes over time for its featuring, managing, viewing or only understanding what happen with itself [5, 6]. For it, in section 2 will be present a general review of procedures to understanding a thermodynamic system as the incubation is; section 3 shows 3 experiments used to analyze

5th IMRMPT IOP Publishing
Journal of Physics: Conference Series **1386** (2019) 012070 doi:10.1088/1742-6596/1386/1/012070

data in thermodynamic system using data-logger, and finally section concluded the experimental results based in statistics metrics gotten for each experiment.

On other hand, Colombia's agriculture is a vulnerable economic sector for who was presented this researching, giving a approach of data analysis in birds' incubation systems, or similar production methods, aiming to improve techniques in industrial businesses with emerging tools like machine learning and computer science. Indeed, materials field involves engineering tools to solve problems related to measuring or featuring systems as much as materials and growing up the methods to study physics processes.

2. Methods and materials

The methodology was divided in forth parts: Design of the prototypes for testers and experiments, where the main issue is make simple as so as be possible; the second part aims to validate data-logger process, using experiments and birds' incubation system; the third part we introduce a controller to the system in order to manage automatically coordinates; and the last part take off data to a analysis into deep learning technique named artificial neural networks (ANN).

2.1. Design and requirements

Focusing to obtain a data analysis of values obtained from sensors installed in a thermodynamic system, thus the framework [7] gives arduino open-source hardware and software as an embedded system at several applications due to three main features:

- Doesn't require external manage to manipulate its behavior or main functions.
- Work in real-time where measurements are done with a sampling time (T_s).
- Includes computational intelligence, that is, the machine takes a decision when an event has occurred.

An embedded system must have the following main facts: reliability, maintainability, and availability such as are implemented for data-logger into code structure, using the paradigm of object-oriented programming and the code is released on electronic repository under open-source Massachusetts Institute of Technology (MIT) license, allowing public use and implementation of the system or modification for specific application or variable, under conditions that there isn't legacy warranty of the system or liability.

Figure 1 shows flow chart for data-logger, since system with temperature and relative humidity (RH) coordinates measured with sensors at the same time that voltage and current are recorded, this information go to board connected with Atmega328 through serial digital communications protocol, where also are connected a panel and micro secure digital (microSD) card memory, finally a serial to USB protocol is used as external monitor for data-acquisition in real-time.

The atmega328 microcontroller process information obtained to carry out data to microSD card memory, this saves the information obtained using comma-separated values (CSV) format with notepad file. On other hand, the panel mentioned previously has the following options:

(i) New file: Close the current file and create a new file using Electrically Erasable Programmable Read-Only Memory (EEPROM) data where will be information about what number is counted until this moment (maximum 255 files).

(ii) Reboot system: Allows overwrite all files saved since log 00 until 254. This action is not recommended when information wasn't saved previously, only is added in case that system has been collapsed by wrong data flow.

5th IMRMPT IOP Publishing
Journal of Physics: Conference Series **1386** (2019) 012070 doi:10.1088/1742-6596/1386/1/012070

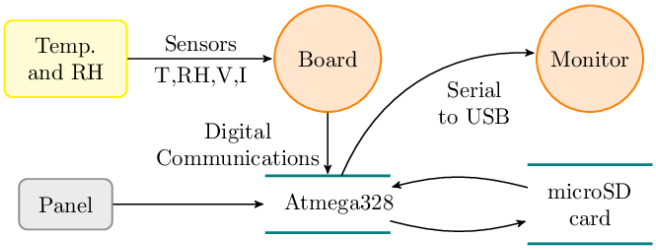


Figure 1. Data flow from thermodynamic system to micro SD storage using atmega328 incorporated on Arduino Board.

2.2. Laboratory proofs

In laboratory proofs has been necessary define which thermodynamic system will test the data-logger behavior [2], for it we've been proposed an incubator system as part of our research, using a system with ceramics resistors, these resistors are known for heating process, because temperature value doesn't affect resistance value significantly [8]. On other hand, was required to define ventilation conditions to the incubator system using the most optimized fan [2], to reduce loss of energy unnecessary at secondary, to determine that device, we evaluate volumetric flow rate describe in Equation (1) using 5 fans with 6 cm of ratio which are implemented in the incubation system to measure air flow velocity with and anemometer, and their energy consumption with measurements of currents supplied when are 12 V in their input.

$$Q = \frac{dV}{dt} = \upsilon * A \tag{1}$$

Table 1 shows the 5 fans used with their respective model in industry, flow velocity rate, current consumption, volumetric flow rate computed with Equation (1), and dQ/dP relationship in m^3/J that represent the level of energy transferred to move a volume.

Being the fan number 1 was selected to the incubation system due to its grades obtained, nevertheless errors' values defined as 1% for anemometer and 4% for ammeter didn't impact to the selection due to computing process were reducing values.

Table 1. Results of test for ventilation conditions.

No.	Model	Speed (m/s)	Current (A)	Q (m^3/s)	$\frac{dQ}{dP}(10^{-2}m^3/J)$
1	Artic PWM PST	$3.50 \pm 1\%$	$0.096 \pm 4\%$	$0.0396 \pm 1\%$	3.4375 ± 0.1719
2	Artic Silent	$2.00 \pm 1\%$	$0.060 \pm 4\%$	$0.0226 \pm 1\%$	3.1389 ± 0.1569
3	BOK BDH1202SS	$4.20 \pm 1\%$	$0.220 \pm 4\%$	$0.0475 \pm 1\%$	1.7992 ± 0.0900
4	Car Fan 12V	$3.00 \pm 1\%$	$0.110 \pm 4\%$	$0.0339 \pm 1\%$	2.5682 ± 0.1284
5	SXD8025S12M	$3.00 \pm 1\%$	$0.230 \pm 4\%$	$0.0339 \pm 1\%$	1.2283 ± 0.0614

2.3. On-off adaptive-predictive controller

The on-off adaptive-predictive controller (OOAPC) is a device that allows manage automatically the input source using conditions desired of temperature, only require two fixed temperatures to connect or disconnect source for heating system, is submitted to characterize system behavior with a certain action of control is applied over thermodynamic system. OOAPC predicts the

5th IMRMPT IOP Publishing
Journal of Physics: Conference Series **1386** (2019) 012070 doi:10.1088/1742-6596/1386/1/012070

temperature that system is going to reach in a determined time of inertia defined as $a_j = 60$ seconds for this case, this value is multiplied with temperature gradient dT/dt to get temperature step and previous temperature is added giving a result for prediction described in Equation (2).

$$P(t + a_j) = a_j \frac{dT}{dt} + T_{(t-1)} \tag{2}$$

To compute gradient is enough take n number of samples of the sensors measurements and obtain mean value over time spent to the sampling procedure as is showed in Equation (3).

$$\frac{dT}{dt} = \frac{1}{n * T_s} \sum_{i=1}^{n} T_i \tag{3}$$

With this prediction value, the controller turn off an electromagnetic relay (ER) to supply energy towards the heating resistor connected at normally closed port of the ER when temperature is 37.0 °C or less; controller turn on the ER when temperature of prediction is 37.5 °C or greater, these two actions are made with assumption of a inertial time fixed by a_j.

2.4. Artificial neural networks modeling

ANNs is a technique of deep learning into a machine learning area included in artificial intelligence science, that allows us to learn about of one system from its data when input and output is known [9]. In literature has been found a relationship between temperature and humidity, defined experimentally as inverse [5, 10–12], that is a tricky trouble into a heating interchange systems and model of this phenomenon is uncertain. Thus, ANN is applied over data recorded using data-logger to emulate system in order to obtain more information about this case and how it's modelled [2], using 10% of samples to train and 90% of samples to test two models of ANNs which are presented in next sections.

2.4.1. Single neuron-layer model. Is a model with a single dense neuron unit into a one layer that is similar to linear regression model where the neuron weight (w) is the slope and neuron bias (b) is the intersection in dependent axis like is described in Equation (4).

$$y = w * x + b \tag{4}$$

The data input (x) is T as the temperature and the output (y) is RH as the relative humidity, obtaining a model $RH(T)$, also a $T(RH)$ model is considered to analyze.

2.4.2. Multiple neurons model. The another model consists in six neurons distributed uniformly at two hidden layers, that is a approach that system isn't linearly fitted as is presented in Figure 2, where hidden layers contains neurons a_0, a_1, a_2 and b_0, b_1, b_2 for hidden layer 1 and hidden layer 2, respectively.

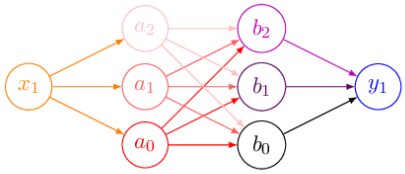


Figure 2. ANN dense with six neurons in two hidden-layers for one input and one output neurons.

That neurons units are dense type of neurons, as well as input layer and output layer have singles neurons defined as x_1 and y_1, respectively. The model for multiple is very complex in

5th IMRMPT IOP Publishing
Journal of Physics: Conference Series **1386** (2019) 012070 doi:10.1088/1742-6596/1386/1/012070

comparison with single neuron-layer model (SNLM), due to how its weights and bias are related. Literature has the assumption that this type of models require much data due to great quantity of variables or change the training mode in order to improve as fast as be possible [13].

3. Experimental results

Three experiments was made to validate our system, where the first consist in the varying of the voltage value for input supply to the heating system, second experiment implements OOAPC and two fixed values in thresh starting in a low temperature environment (transient state), and in the third experiment OOAPC remains under steady state conditions with a closest temperature to the desired value.

3.1. Experiment number 1

Starting experiment sessions, the system was undergoing to conditioner air flow of 18.0 °C until reach 25.0 °C in the system, these air flow remains during all time that experiment spent. In Figure 3 is showed the temperature and RH curves when input voltage is manually varied from 2 V to 12 V in steps of 2 V as is presented in curves of Figure 4, this task is repeated 3 times over 30 minutes of execution. Red curve is the temperature on heating source and cyan curve correspond to its RH, orange curve is the temperature at load zone and blue curve its RH. In another plot green curve is voltage in electrical energy source to run all devices included in the system, while purple curve is the current consumed, this curves have critical behaviors for source zone but load trends to linear behavior and soft manipulation, when energy supply seems has problems with maximum voltage.

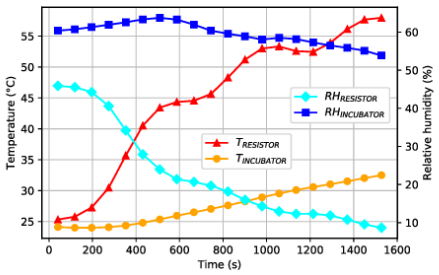


Figure 3. Temperature and RH curves at supply zone and load zone for experiment 1.

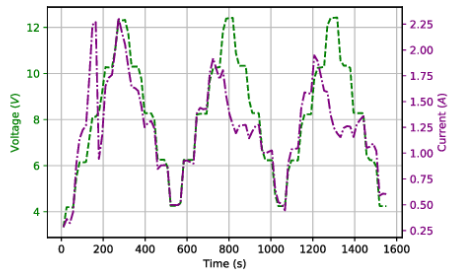


Figure 4. Voltage and current curves at energy supply for experiment 1.

The next has been apply the neural networks models SNLM and multiple neurons model (MNM) using 695 samples to train and 6257 samples to test both models, the training is done using mean squared error (MSE) with Adam Optimizer fixed in 0.1 through 500 epochs, giving the linear Equation $RH(T) = -0.9066 * T + 83.9731$, where temperature is in Celsius degrees and RH is % units, this linear Equation is obtained from SNLM model that reaches 1.9163 of MSE and 1.1389 of mean absolute error (MAE). The MNM model improved grades respect to SNLM obtaining 1.8397 of MSE and 1.0961 of MAE.

3.2. Experiment number 2

Second experiment implements OOAPC device with its measuring point located in load zone, source zone doesn't represent a reliable place to set temperature desired. How was looked at experiment 1, colors in curves represent the same variables, but is evident a trending to a defined value for Figure 5 in temperature values, remaining softest to load zone and some variant for

5th IMRMPT IOP Publishing
Journal of Physics: Conference Series **1386** (2019) 012070 doi:10.1088/1742-6596/1386/1/012070

source zone while RH curves have similar inverse behavior. In the Figure 6 current a voltage have a mid-point until desired temperature is near, when these event occurs, curves trends to oscillate creating a quasi-periodic signal to establish the desired temperature.

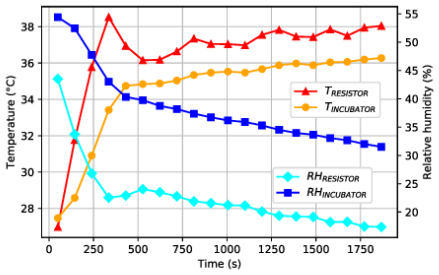


Figure 5. Temperature and RH curves at supply zone and load zone for experiment 2.

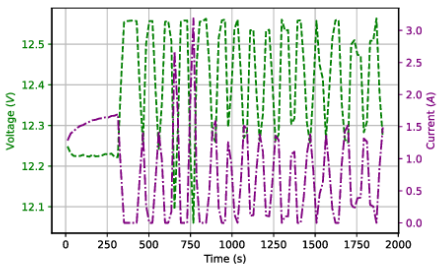


Figure 6. Voltage and current curves at energy supply for experiment 2.

For this experiment, data analysis using ANN models gives the SNLM model better than MNM model due to the metrics applied, MSE for SNLM was 2.2857 when MNM gave 5.1834 and MAE was 1.2820 for SNLM and 1.9001 for MNM, that is, SNLM model is almost 44.10% more accuracy than MNM model for transient state. Nevertheless this experiment used only 838 samples to train and 7551 to test giving a linear regression model described in Equation (5) and using the same parameters for training mentioned in the experiment 1.

$$RH(T) = -2.5105 * T + 125.2680 \tag{5}$$

3.3. Experiment number 3

This experiment aims to steady state behavior, when desired temperature has been reached by OOAPC and the importance is find how much varies its values like Figure 7 where has been seen a quasi-static RH and temperature trending to mid-point as much as the time pass, getting a quasi-periodic signal in Figure 8 to maintain the desired conditions.

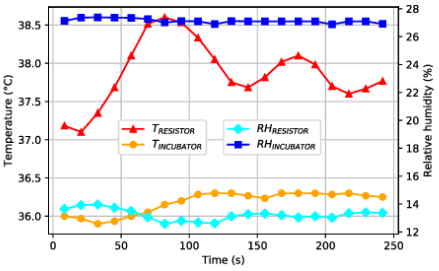


Figure 7. Temperature and RH curves at supply zone and load zone for experiment 3.

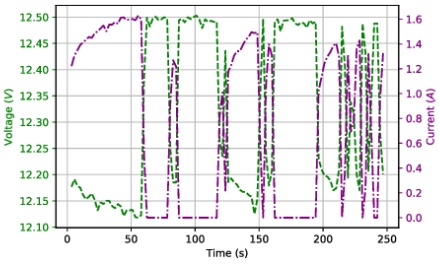


Figure 8. Voltage and current curves at energy supply for experiment 3.

Finally, for steady state was used 108 samples to train and 980 samples to test remaining the training parameters from the previous experiments, but the metrics was significantly reduced for both models, where MSE gave 0.2661 for SNLM model and 0.6111 for MNM model, MAE

gave 0.4661 for SNLM model and 0.0196 for MNM model. Indeed, SNLM model remains as the better model for two or three experiments, denoting what OOAPC controller may be the difference for these results, trending the system to a linear regression such as for this case is$RH(T) = 0.7192 * T + 0.7029$, this Equation is very different respect to Equations (**??**) and (5), in this experiment RH didn't vary notably that can generate this great difference and giving a uncertain hypothesis about of temperature and RH are related.

4. Discussion

The data-logger system is robust and embedded device to acquire data from real environment in order to process that data using open-source computational tools like jupyter notebook, RStudio or Spyder [5]. Real systems must be delimited before laboratory proofs, this gives initial conditions to improve results and make it more comparable with another works in literature likes [3, 6, 14]. Sensors cannot read in real-time, they ever delay a defined time to acquire and process the values obtained, for it is necessary assume predictions over them, that can make more adaptive for time-variant systems with certain parameter of setting for its adaptability supported on data sheets' parameters like sampling frequency or sensibility. Data analysis using ANN tools got interesting results, where in three different environments SNLM and MNM models responded with high accuracy using a relation 1:9 for cross validation, that is, SNLM and MNM are models optimal for time-variant systems as was purposed by [2].

5. Conclusion

With the results of experiment 1, was obtained a average power value in 10.9681 W for 1570 seconds spent such as the energy consumption was 17.210 KJ when energy supply is manually varied, implicating high variant behaviour for the system and requiring MNM to improve accuracy in its modelling. With the results of experiment 2 was obtained a average power value in 9.7136 W for 1918 seconds spent such as the energy consumption was 18.630 KJ when energy supply is in transient state using OOAPC controller. With the results of experiment 3 was obtained a average power value in 9.4818 W for 250 seconds spent such as the energy consumption was 2.370 KJ when energy supply is in steady state using OOAPC controller. Both experiment 2 and 3 demonstrate high performance for OOAPC when it optimizes energy consumption while systems trend to linear model or SNLM was a near representation for itself.

References

[1] Wang J 2009 *Progress in Natural Science* **19(1)** 125
[2] Reyes-Rosas A *et al.* 2012 *Revista Chapingo. Serie horticultura* **18(1)** 125
[3] Fernandes M, Canito A, Bolón-Canedo V, Conceição L, Praça I and Marreiros G 2019 *International Journal of Information Management* **46** 252
[4] Sousa R, Pereira M, Pereira F M Q and Araujo G 2019 *Journal of Parallel and Distributed Computing* **130** 126
[5] Schito E, Pereira L D, Testi D and da Silva M G 2019 *Data in Brief* **24** 103788
[6] Nayak P, Mukherjee A K, Pandit E and Pradhan S K 2018 *Rice Science* **25(1)** 1
[7] González Morales J *et al.* 2017 *Diseño e implementación de un control de temperatura y humedad para un prototipo de incubadora artificial de pollos* (Cali: Pontificia Universidad Javeriana)
[8] Contreras Contreras G F and Dulcé-Moreno H J 2018 Diseño y construcción de una incubadora de aves de bajo consumo energético *5th International Week of Science, Technology and Innovation* (San José de Cúcuta: Universidad Francisco de Paula Santander) p 235
[9] Matich D J 2001 *Redes neuronales: Conceptos básicos y aplicaciones* (Rosario: Universidad Tecnológica Nacional)
[10] Laukkarinen A and Vinha J 2017 *Energy Procedia* **132** 711
[11] Sallam G A and Elsayed E 2018 *Ain Shams Engineering Journal* **9(1)** 1
[12] Xu X, Zhong Z, Deng S and Zhang X 2018 *Energy and Buildings* **162(1)** 163
[13] Xu X, Zhong Z, Deng S and Zhang X 2019 *Materials & Design* **162** 300
[14] Bou-Llusar J C and Satorra A 2019 *BRQ Business Research Quarterly* 1

Anexo 4. Trabajo escrito aceptado para una publicación especial de la Sociedad Colombiana de Ingenieros

PREDICCIÓN DEL NIVEL DE CONTAMINACIÓN ELECTROMAGNÉTICA UTILIZANDO GOOGLE MAPS Y TÉCNICAS IDW BASADO EN MEDIDAS

Pedro Andrés Beltrán Ortega, Ghiordy Ferney Contreras Contreras
Universidad Francisco de Paula Santander

Temática: Energía _____ Agroindustria _____ Saneamiento _____ Materiales ____ Procesos Productivos y Logísticos _____ Tecnologías 4.0 _X_

Justificación de temática: Este proyecto está enmarcado en el tópico de tecnologías 4.0 debido a que su propósito principal es predecir los niveles de contaminación electromagnética usando como herramientas los datos geo referenciados de una campaña de medición, un aplicativo web para dispositivos móviles y de escritorio, y finalmente una técnica de interpolación basada en el inverso de la distancia ponderada. El producto final del trabajo es un sistema de información, donde se almacenan los datos de la campaña y se realizan las predicciones a partir de estos, además de plasmar la información en mapa de calor sobre el mapa de Google.

RESUMEN: Este trabajo presenta el desarrollo de un aplicativo web y móvil, la cual predice el nivel de contaminación electromagnética (CE) basado en una campaña de medidas tomadas en el campus universitario de la Universidad Francisco de Paula Santander (UFPS) utilizando la plataforma de Google Maps y la interpolación con la distancia inversa interpolada (IDW), permitiendo a la comunidad cucuteña disponer de los valores de CE a los que están expuestos y las recomendaciones de la Unión internacional en Telecomunicaciones (UIT) que deben asumirse para proteger la salud de las personas expuestas. Con esto, el campus UFPS está libre de CE.

INTRODUCCIÓN: El auge de la industria electrónica y los avances científicos de los últimos años, han generado importantes cambios en la vida humana y en todos los campos del conocimiento, la producción masiva de dispositivos electrónicos responde al ritmo de consumo de los usuarios, hoy en día se encuentran dispositivos electrónicos en todas partes, su uso se ha extendido desde los sistemas de control procesos industriales y científicos hasta la aplicación de Internet de las Cosas en los hogares de centros urbanos, en las que predominan los dispositivos de comunicación inalámbrica. Dado que los dispositivos electrónicos producen un campo eléctrico a su alrededor, ha surgido preocupación sobre el impacto que dicho campo eléctrico pueda causar sobre el cuerpo humano y se ha concebido el concepto de contaminación electromagnética dentro de la contaminación ambiental (Şahin, 2014). A Nivel internacional La Comisión Internacional para la Protección de la Energía No Ionizante (ICNIRP) estableció límites seguros de exposición a la radiación electromagnética no ionizante (RNI), los cuales son adoptados por La Unión Internacional de Telecomunicaciones (UIT), bajo la recomendación UIT-T K.52, la cual fue adecuada por el gobierno de Colombia en el Decreto 195 de 2005 como norma de cumplimiento nacional (Rodríguez, 2010). Con el objeto de verificar el cumplimiento de la recomendación internacional UIT-T K.52 en el campus de la sede central de la Universidad Francisco de Paula Santander (UFPS), se realizó una campaña de medición de campos electromagnéticos, con el propósito de conocer y cuantificar los niveles campos electromagnéticos (CEM) existentes. Donde los resultados, son expuestos en una aplicación web que contiene un mapa de calor geográfico de Google Maps, permitiendo a la comunidad verificar los niveles de contaminación electromagnética de forma clara y

concisa, a los que se está expuesta.

DESCRIPCIÓN DEL PROBLEMA: La falta de claridad y/o el desconocimiento de la población acerca de los CEM y sus posibles efectos en la salud, ha generado gran preocupación entre las personas debido al avance tecnológico en comunicaciones inalámbricas, el incremento de su infraestructura y la correspondiente asignación de frecuencias en bandas superiores a las utilizadas actualmente por los prestadores de servicio de telecomunicaciones, exigiendo a estas empresas el despliegue de muchos transmisores localizados en distancias cortas que pueden elevar el nivel de radiación presente en el ambiente, causando conflictos entre la comunidad y las empresas prestadoras de los servicios de comunicaciones inalámbricas y energía eléctrica.

OBJETIVOS:

OBJETIVO GENERAL

Predecir los niveles de contaminación electromagnética utilizando Google Maps y técnicas de interpolación sobre medidas.

OBJETIVOS ESPECÍFICOS

- Realizar campaña de medidas de radiación del campo electromagnético geo referenciadas dentro del campus de la Universidad Francisco de Paula Santander.
- Seleccionar y Aplicar una técnica de interpolación para la predicción de niveles de campo electromagnético basado en medidas.
- Diseñar una aplicación que permita el almacenamiento y la visualización de los niveles de contaminación a través de un mapa digital de Google Maps.

METODOLOGÍA: Desde su planteamiento se establecieron tres objetivos que en conjunto conforman el eje sobre el cual se ha desarrollado el trabajo. Para ellos, se estableció una metodología secuencial, donde cada objetivo soporta una serie de actividades. Inicialmente se realizó la campaña de medición en banda ancha (100 kHz – 3 GHz) siguiendo la recomendación UIT.T K83. Para la medición de la banda ancha, el Grupo de Investigación y Desarrollo en Electrónica y Telecomunicaciones (GIDET) de la UFPS, proporcionó los siguientes componentes: Sonda de banda ancha (sonda WPF3), Instrumento de medición (Medidor de campo electromagnético SMP2) y Soporte no reflectante (Trípode de madera).

Posteriormente de tener una base sólida de valores almacenados, es posible utilizarla como referencia para interpolar los resultados, y poder estimar el valor medido para un punto arbitrario, de esta manera, es posible mostrar al usuario el estimado de cualquier punto que él mismo seleccione en el mapa de calor dejado en el aplicativo web y móvil. Dado que el trabajo actual se realiza sobre un espacio geográfico establecido en un mapa, se tienen en cuenta métodos de interpolación que se ajustan a una medición terrestre. Por eso, se analizaron los principales métodos de interpolación que se utilizan en sistemas de información geográfico, entre esos métodos hay algunos probabilísticos como Kriging, y otros determinísticos como la distancia inversa ponderada (IDW).

Además de la metodología en la adquisición de los datos y su posterior predicción para ubicaciones sin medición, es necesario definir también una metodología de desarrollo de software que se adapte a las necesidades del trabajo. En general, se consideró construir la aplicación progresivamente, usando las fases de: diseño, desarrollo, pruebas y entrega; generando pequeñas iteraciones en cada una, de las cuales se desarrollan nuevas funcionalidades, que son diseñadas en base a requerimientos técnicos-funcionales y sometidas a pruebas antes de pasar a la fase de entrega al público.

RESULTADOS: El objetivo general de este trabajo es predecir los niveles de contaminación electromagnética utilizando Google Maps, y como parte del proceso se han obtenido seis resultados asociados a la predicción de dicha contaminación que pueden enmarcarse en:

- La campaña de mediciones de campo electromagnético en el campus de la UFPS donde se realizaron 244 mediciones, con el propósito de conocer y cuantificar los niveles campos

electromagnéticos existentes. La campaña de medición se desarrolló para la medición de la banda ancha comprendida entre las frecuencias de 100 kHz – 3 GHz, siguiendo la recomendación UIT.T K83.

- Los niveles de exposición a campos electromagnéticos, en el cual, el valor máximo medido fue inferior a los límites máximos de exposición, en un valor del 96.5% para la zona del Campus UFPS.
- La aplicación web con acceso público, dejada a disposición de la región para la visualización y verificación de los niveles de contaminación electromagnética, donde adicionalmente es un soporte para nuevas campañas de mediciones sobre la radiación no ionizante en la ciudad de San José de Cúcuta.
- La base de datos que almacena toda la información, insumo fundamental para el debido funcionamiento de la aplicación; la aplicación web, a través de la cual cualquier usuario puede acceder y verificar los niveles de contaminación electromagnética de forma clara y concisa, a los que se está expuesta, incluye gráficos que permiten visualizar el comportamiento de cada una de las mediciones realizadas, comparándolo con el límite máximo permitido en Colombia.
- El mapa de calor geográfico de Google Maps, contenido en la aplicación web, donde los valores tomados para una variable en un mapa de dos dimensiones son representados como colores en una escala, que en última instancia es el punto de acceso a la visualización de los niveles de contaminación electromagnética.
- La interpolación de las mediciones a través de la técnica IDW, gracias a la cual se predice el valor del campo electromagnético en cualquier otro punto en el campus que el usuario desee e indique
- Documentación de la aplicación, correspondiente a todo el contenido documental técnico del software, manuales, ficha técnica, entre otros, permitiendo una mayor claridad de su funcionamiento y construcción, de manera de que sea replicable y factible para futuras implementaciones en otros trabajos.

CAMPO DE APLICACIÓN:

- Sistemas de Comunicaciones que usen el espectro electromagnético.
- Técnicas de predicción para mediciones geo referenciadas.
- Análisis de radiación electromagnética en medios isotrópicos.
- Salud en áreas sometidas a dosis considerables de radiación no ionizante.

CONCLUSIONES: El desarrollo de este trabajo ha permitido conocer a profundidad los niveles de contaminación electromagnética, medidos en banda ancha en el campus de la Universidad Francisco de Paula Santander. A raíz de estas mediciones, ha sido posible concluir que el valor máximo medido de RNI fue inferior a los límites máximos de exposición, en un valor del 96.5% por fuera de la zona ocupacional en donde están emplazados los transmisores, reglamentado en el decreto 195 del MITIC y por la UIT en la recomendación internacional UIT-T K.52. Las zonas con mayor valor medido de RNI se encuentra en: la cancha de césped de futbol de la UFPS (7.89949588N, -72.48747106W), con un valor medido de 0.9851 V/m y alrededor del edificio "La casona" (7.895894N, -72.488224W) con un valor medido de 0.9494 V/m. Cabe resaltar que estos valores se encuentran por debajo del límite máximo establecido y no presenta riesgo alguno para la comunidad. El 73.77% de los valores medidos de campo eléctrico en el campus de la UFPS se encuentran entre los rangos de 0.2473 V/m a 0.4603 V/m. así mismo, solamente el 2.86% de los valores medidos de campo eléctrico se encuentran entre los rangos de 0.6733 V/m a 1 V/m. Por lo

que el valor promedio de RNI medido en toda la UFPS fue inferior un 98.6% del límite máximo establecido, con lo cual se puede deducir que no hay contaminación electromagnética significativa en el campus de la UFPS. Una mayor cantidad de puntos de entrada considerados para la implementación de los métodos de interpolación no mejora la precisión. Es decir, los valores de campo eléctrico que están más cercanos de la ubicación de donde se realiza la predicción tienen mayor correlación espacial. Por este motivo, se llevó a cabo la interpolación con los 10 valores de campo eléctrico más cercano obteniendo así un mínimo error promedio (0.0587452 V/m) que tomando una mayor cantidad de valores cercanos. De los modelos de interpolación analizados el que permitió la mejor predicción de los valores de campo eléctrico, obtuvo un menor tiempo de procesamiento y contó con un menor error promedio respecto a los valores medidos con los 10 valores de campo eléctrico más cercano, fue el método de interpolación por el inverso de la distancia (IDW) con un error promedio del 0.0587452 V/m, en comparación con los modelos matemáticos del método de interpolación Kriging, para el modelo de Kriging lineal (0.077881893 V/m), el modelo de Kriging esférico (0.066242387 V/m), el modelo de Kriging exponencial (0.076292593 V/m) y el modelo de Kriging gaussiano (0.076292593 V/m).

BIBLIOGRAFÍA:

[1] ICNIRP, "Recomendación para limitar la exposición a campos eléctricos, magnéticos y electromagnéticos (hasta 300 GHz)," 1998.
[2] UIT, "Recomendación UIT-T K.52 (Orientación sobre el cumplimiento de los límites de exposición de las personas a los campos electromagnéticos)," 2018.
[3] UIT, "Recomendación UIT.T K.83 (Supervisión de los níveles de intensidad del campo electromagnético)," 2011.
[4] Ministerio de Comunicaciones. (2005). Decreto 195 de 2005
[5] Agencia Nacional del Espectro. (2016). Resolución N°711 de 11 OCT. 2016. Bogotá: Despacho Director General.
[6] Organización Mundial de la Salud. (2018). Campos electromagnéticos. Recuperado de: http://www.who.int/peh-emf/about/WhatisEMF/es/
[7] Rodríguez, C. (2010). Estudio de los niveles de radiación electromagnética no ionizante en varias zonas de la ciudad de Bucaramanga. UIS Ingenierías, 9(2), 1.
[8] Şahin, M. (2014). Electromagnetic pollution measurement in the RTE university campus area.
[9] Rodríguez, M. (2013). Medición de radiación electromagnética no ionizante como un servicio de telecomunicaciones. Radio Gis Grupo de investigación telecomunicaciones - Universidad Industrial de Santander, Bucaramanga – Colombia.
[10] García, W. (2014). Evaluación de niveles de exposición electromagnética causados por las estaciones base de telefonía móvil celular. Manizales: Universidad de Manizales
[11] Google Maps Platform. (2018). Maps JavaScript API (Heatmap layer). Recuperado de: https://developers.google.com/maps/documentation/javascript/heatmaplayer
[12] ArcGIS for Desktop. (2004). Cómo funciona IDW. Recuperado de: http://desktop.arcgis.com/es/arcmap/10.3/tools/3d-analyst-toolbox/how-idw-works.htm
[13] ArcGIS for Desktop. (2016). Cómo funciona Kriging. Recuperado de: http://desktop.arcgis.com/es/arcmap/10.3/tools/3d-analyst-toolbox/how-kriging-works.htm
[14] WaveControl. (2018). Página principal. Recuperado de: https://www.wavecontrol.com/rfsafety/en

Anexo 5. Certificado de ponente en la 5th International Week of Science, Technology & Innovation

5th INTERNATIONAL WEEK OF SCIENCE TECHNOLOGY & INNOVATION NOVEMBER 20 - 23, 2018

EVENTO INTERNACIONAL SIES+

LA ALIANZA SISTEMA UNIVERSITARIO DE NORTE DE SANTANDER SIES+

CERTIFICA QUE:

GHIORDY FERNEY CONTRERAS CONTRERAS
C.C. 1090503143

Participó como **PONENTE** en modalidad **POSTER** en el evento **5th International Week of Science, Technology & Innovation** que se desarrolló del 20 al 23 de noviembre del año 2018 en San José de Cúcuta, Norte de Santander Colombia, con el tema "CLUSTER CV2: A COMPUTER VISION APPROACH TO SPACIAL IDENTIFICATION OF DATA CLUSTERS".

COORDINADOR MESA DE INVESTIGACIÓN ALIANZA SIES+
VICERRECTOR ASISTENTE DE INVESTIGACIÓN
UNIVERSIDAD FRANCISCO DE PAULA SANTANDER

SECRETARIA TECNICA MESA DE INVESTIGACIÓN ALIANZA SIES+
DIRECTOR DE INVESTIGACIÓN
UNIVERSIDAD DE SANTANDER – CAMPUS CÚCUTA

Anexo 6. Certificado de ponente en el XXII International Symposium on Image, Signal Processing and Artificial Vision

Anexo 7. Certificado de ponente en el 5thInternational Meeting for Researchers in Materials and Plasma Technology

Anexo 8. Certificado de ponente en el evento I Muestra Nacional de Proyectos de Pregrado en Ingeniería ''Expresa tu ingenio''

EXPRESA TU INGENIO
I MUESTRA NACIONAL DE PROYECTOS DE PREGRADO EN INGENIERÍA

SOCIEDAD COLOMBIANA DE INGENIEROS

ACOFI
Asociación Colombiana de Facultades de Ingeniería

La Sociedad Colombiana de Ingenieros – SCI y
la Asociación Colombiana de Facultades de Ingeniería – ACOFI

Certifican la participación de

Ghiordy Ferney Contreras Contreras

En el Proyecto

PREDICCIÓN DEL NIVEL DE CONTAMINACIÓN ELECTROMAGNÉTICA UTILIZANDO GOOGLE MAPS Y TÉCNICAS IDW BASADOS EN MEDIDAS

De la Temática *Tecnologías 4.0*

Presentado en la

I MUESTRA NACIONAL DE PROYECTOS DE PREGRADO EN INGENIERÍA "EXPRESA TU INGENIO"

GERMAN PARDO ABARRACÍN
PRESIDENTE
Sociedad Colombiana de Ingenieros

CARLOS ARTURO LOZANO
PRESIDENTE
Asociación Colombiana de Facultades de Ingeniería

Dado en Bogotá, D.C., a los 23 días del mes de agosto de 2019