RECONOCIMIENTO DE EXPRESIONES FACIALES USANDO UN SISTEMA EMBEBIDO

SERGIO DANIEL HERNÁNDEZ REYES

JUAN PABLO ZULUAGA CALVACHE

PONTIFICIA UNIVERSIDAD JAVERIANA

FACULTAD DE INGENIERÍA

PROGRAMA DE INGENIERÍA ELECTRÓNICA

BOGOTA, COLOMBIA

2022

RECONOCIMIENTO DE EXPRESIONES FACIALES USANDO UN SISTEMA EMBEBIDO

SERGIO DANIEL HERNÁNDEZ REYES

JUAN PABLO ZULUAGA CALVACHE

TRABAJO DE GRADO PRESENTADO PARA OPTAR AL TÍTULO DE INGENIERO ELECTRÓNICO

DIRECTOR:

JAIRO ALBERTO HURTADO LONDOÑO Ph.D

PONTIFICIA UNIVERSIDAD JAVERIANA

FACULTAD DE INGENIERÍA

PROGRAMA DE INGENIERÍA ELECTRÓNICA

BOGOTA, COLOMBIA

2022

Dedicatoria

"Este trabajo de grado se lo dedico a mis padres por el apoyo y sabiduría de sus consejos que al pasar del tiempo me han formado como ser humano. Por lo cual, siempre serán la mayor motivación para cumplir mis sueños".

Sergio Daniel Hernández Reyes

Dedicatoria

"Este trabajo de grado se lo dedico a mis padres por su constante apoyo, a mi hermano que fue quien inicialmente me inspiró a iniciar mis estudios como ingeniero"

Juan Pablo Zuluaga Calvache

Agradecimientos

Agradezco a todos aquellos que han estado presentes en la realización de este trabajo de grado, agradezco a Sebastián Mariño, Andrés Barreto y Daniel Barandica personas las cuales aprecio mucho y trabajé durante toda la carrera con ellos, especial agradecimiento a mi compañero Sergio Hernández que además de ser compañeros de trabajo de grado tenemos una gran amistad, a Sofía Morales que siempre estuvo en todo este proceso y fue un gran apoyo en los momentos más complicados tanto académicos como personales, Agradezco inmensamente al ingeniero Jairo Alberto Hurtado por su disposición en el acompañamiento de este trabajo de grado y durante toda la carrera, finalmente agradezco a mi familia quienes han sido mi principal motivación para seguir adelante a lo largo del camino.

Juan Pablo Zuluaga Calvache

Agradezco a la Pontificia Universidad Javeriana y especialmente al ingeniero Jairo Alberto Hurtado Londoño por su disposición para acompañarnos durante todo este proceso; persona la cual apreciamos y admiramos sus principios, su valor, su amplio conocimiento y capacidad de enseñanza. Agradezco a mis padres y hermanos por su apoyo, sus consejos me ayudaron a seguir adelante en mis años de estudio, con el esfuerzo que se merecen por darme todas las comodidades con el fin de que pueda ser un profesional que dedique su vida en ayudar al prójimo.

Sergio Daniel Hernández Reyes

Resumen

La detección facial a través de imágenes captadas por cámara es una necesidad tecnológica que al pasar del tiempo ha sido mejorada con la aplicación de algoritmos, el aumento de resolución en los dispositivos que capturan el rostro, la velocidad de procesamiento mejorada por los nuevos procesadores y tarjetas gráficas que existen en el mercado. Este documento busca exponer al lector acerca de diferentes formas que existen actualmente para lograr identificar rostros en tiempo real y descifrar la expresión facial reconocida en dichos rostros en diferentes sistemas embebidos, por lo que demuestra que no es necesario poseer un equipo con *hardware* costoso o las mejores especificaciones para lograr dicho fin, se exponen los resultados del sistema funcional compilado en PC, *Raspberry pi* 4 y *Jetson Nano*.

La identificación de rostros y sus diferentes emociones es un campo de investigación amplio, por esta razón se decidió acotar las variables que afectan la población, tamaño de muestra e instrumento de recolección de datos utilizado. La edad de las personas que se quiere identificar el rostro es de 16 a 45 años, mujeres y hombres incluidos en su funcionamiento (sin diferencia en su funcionalidad), detección de diferentes personas a la vez bajo diferentes escenarios como la variación de luz, enfoque y ángulo de inclinación en el rostro.

Al implementar el código funcional en los tres sistemas embebidos nombrados anteriormente se identificó que el rendimiento depende de qué tan robusto sea el *hardware* sobre el que funciona el *software*, debido a que en un PC con un procesador de frecuencia base 3,9 GHz y tarjeta gráfica integrada de 1900 MHz el *stream* de la cámara e interfaz de usuario se nota fluido sin latencias ni caída de cuadros por segundo; mientras que, en los otros dos sistemas, aunque es funcional se notó un rendimiento desmejorado del sistema, bajaron los *frames* por segundo y se notaban latencias en el *stream* de la cámara de hasta cuatro segundos, esto sin hablar que al tratar de hacerlo funcionar hubo problemas con la instalación de librerías, funcionamiento del sistema operativo, reconocimiento de la cámara USB e incompatibilidad de versiones de *Python*. La solución a estos problemas está expuesta en el capítulo 4 nombrado “Migración de la solución a los sistemas embebidos”.

La funcionalidad y rendimiento de la predicción de los algoritmos de inteligencia artificial puede variar por diferentes factores como el entrenamiento de la base de datos, el tipo de algoritmos utilizados, los parámetros del algoritmo y en el caso específico de este trabajo de grado puede variar por género, la cantidad de emociones que se puedan identificar y el sistema embebido en el que esté funcionando el programa, con el fin de exponer estas métricas consultar el capítulo 4 nombrado “Resultados”.

Índice general

Índice de figuras

Capítulo 1

1.1 Introducción

La interacción entre la máquina y el ser humano se ha hecho común en los últimos tiempos, entre los dos tipos de comunicaciones que tipifican los comportamientos del ser humano, la comunicación verbal y no verbal; se han visto diferentes sistemas que pueden llegar a traducir lo que decimos, pero no hay muchos que identifiquen, lo que las personas expresan inconscientemente a través de sus expresiones faciales, solo unos pocos trabajos que se encuentran en la literatura con fines académicos pero productos reales hay pocos. Haciendo uso de un sistema embebido y una cámara que capte el rostro del usuario, se quiere llegar a dar una retroalimentación en tiempo real de las emociones a través de la identificación expresiones faciales.

En este proyecto de grado se tiene como fin comparar las diferentes maneras de cuantizar las principales expresiones faciales humanas divididas en estados independientes; variando parámetros como los algoritmos a utilizar, el tipo de sistema embebido sobre el cual correrá el programa, la base de datos que se utilizará para dar una predicción y las diferentes formas de estructurar los procesos dentro de un código de programación.

El siguiente documento consta de la descripción de la problemática, donde se explica el propósito del proyecto y el problema a solucionar. Posteriormente se encuentran los objetivos donde se explican puntualmente los logros alcanzados durante este proyecto de investigación. Continuando con la revisión a la literatura de donde se tomaron referencias para tener una base sólida de donde partir para el desarrollo del prototipo, finalmente se tiene el marco teórico y requerimientos, además de aspectos técnicos y planeación de actividades acerca de la solución.

1.2 Objetivos

1.2.1 Objetivo General

- Desarrollar, haciendo uso de un computador de placa reducida, un sistema de detección de expresiones faciales en tiempo real.

1.2.2 Objetivos Específicos

- Obtener una base de datos de entrenamiento y validación del sistema.
- Comparar diferentes algoritmos para la detección de rostros.
- Comparar diferentes algoritmos para la detección de expresiones faciales.
- Adaptar los algoritmos para los sistemas embebidos.
- Validar funcionamiento del producto respecto a analítica del software desarrollado por la compañía visual Recognition eMotion.

1.3 Requerimientos

- El sistema tiene que reconocer emociones de mínimo dos rostros simultáneamente.
- La información adquirida debe ser transmitida y usada para un análisis.
- Se deben identificar mínimamente dos expresiones faciales
- La población objetivo son los adultos.
- Se debe identificar la expresión facial así la persona se encuentre de perfil.
- Entrada que indique al sistema que inicie el proceso.

Capítulo 2

2.1 Marco teórico

2.1.1 Técnicas de extracción de características para el reconocimiento de expresiones faciales

"En la comunicación no verbal, la expresión facial es importante, pues permite evidenciar expresiones emocionales y estados de ánimo. A través de esta se puede regular la interacción y reforzar al receptor" [11]. Para llegar a reconocer a fondo una expresión facial cerebro humano realiza una extracción de características del rostro de la persona que estamos analizando, esto en el proceso cognitivo del ser humano es simple ya que el cerebro es el que se encarga de todo el trabajo, cuando se busca caracterizar se debe acudir a "las técnicas más usadas para esta tarea, como: seguimiento de movimientos, análisis espacial estadístico, y análisis espacio/frecuencia" [6]

- Análisis dinámico (flujo óptico): Esta técnica tiene dos alternativas: calculando gradientes o realizando segmentación [6], la más usada es la de cálculo de gradientes la cual consiste en determinar los cambios de intensidad de la imagen a niveles de grises, dicho análisis se realizaba en dos planos, sobre el plano de la imagen y sobre el plano del tiempo, "los algoritmos que determinan el flujo a partir del gradiente se basan en los trabajos de *Horn and Shunck*" [6] quienes propusieron una forma de determinar el movimiento de una imagen basándose en que la intensidad de un pixel es constante y este no cambia en el plano del tiempo .

Otra técnica existente es el método Singh y este es muy usado ya que es sensible a movimientos sutiles y este permite extraer dichas características difíciles de percibir a simple vista, "el autor calcula el flujo óptico fusionando dos estimaciones cada una acompañada por una medida de confianza dada en términos de matrices de covarianza" [6]

Figura 1. flujo óptico por el método Singh tomado de [6]

- Análisis en el dominio espacial mediante métodos estadísticos.
- PCA: "La idea del PCA es encontrar la base de vectores que mejor exprese la distribución de las imágenes de las caras dentro del espacio completo" [6] Este procedimiento es con el fin de encontrar los componentes ortogonales de un vector en un espacio específico, al multiplicar cada conjunto de vectores por el conjunto de imágenes de prueba y de entrenamiento se obtiene finalmente un vector de pesos, "que corresponde a la proyección ortogonal de cada rostro sobre las caras propias y se pueden utilizar como características para diferenciar los distintos tipos de rostros y sus rasgos encontrando la menor distancia Euclidiana entre los distintos vectores" [6].
- CA: El ICA es una generalización del previamente mencionado PCA y a diferencia de este, el ICA provee una mejor representación y lectura de la imagen ya que es sensible a dependencias estadísticas de alto orden y no sólo de la covarianza como lo hace el PCA.

- Análisis en el dominio espectral mediante transformadas tiempo – frecuencia.
- Transformada de Wavelet de enteros a enteros: La transformada Wavelet de enteros a enteros consiste en una descomposición polifásica de los bancos de filtros asociados a la transformada Wavelet, "Se agregan operadores de redondeo a la salida de cada etapa del

esquema de actualización, los cuales se pueden desplazar hasta la salida de la actualización y de la predicción" [6]

- Transformada Gabor: "La transformada de Gabor es una modificación a la de Fourier para aplicarla en forma localizada, utilizando una función ventana gaussiana, con lo que se obtiene una transformada muy similar a la transformada wavelet. Para el caso de las imágenes se utiliza una función base que es una exponencial compleja y una función ventana gaussiana en 2D multiplicadas" [6]

2.1.2 Reconocimiento facial

Se define que hay seis tipos de emociones básicas (angustia, disgusto, miedo, felicidad, tristeza y sorpresa) las cuales son universales entre seres humanos. El reconocimiento de expresiones faciales ha sido un tema de estudio por décadas, muchos de los sistemas desarrollados muestran comportamientos no esperados como: la identificación de rostro en sombras e imágenes del ambiente en aplicaciones prácticas debido a la inadaptabilidad de las condiciones en las cuales el sistema fue desarrollado o falta de entrenamiento en los algoritmos de inteligencia artificial. [8]

El reconocimiento facial de la expresión emocional es la capacidad de la mayoría de los seres humanos en reconocer las expresiones básicas, las cuales aparecen en el rostro de las personas. Al experimentar una emoción hay diferentes maneras de las cuales podemos comunicar y expresar aquello que se está sintiendo, estos medios incluyen patrones fisiológicos y conductuales, como los son los cambios faciales, gestuales y del lenguaje que se ocupa a la hora de expresarse. "A través de la expresión facial es posible mostrar estados emocionales específicos, aportando de esta manera información motivacional y de comunicación" (Anguas-Wong y Matsumoto, 2007).[8]

2.1.3 Expresión facial

¿Cómo medir las emociones?

En el campo de la psicología se han empleado dos formas para obtener reportes de una experiencia emocional: el enfoque de emociones discreto y el enfoque dimensional [12]. El primero categoriza las emociones como estados independientes uno de otro, esto acondiciona el sistema con el fin de que los resultados sean patrones de comportamiento únicos. Muchos investigadores prefieren crear categorías de expresiones faciales que son relevantes para un contexto de investigación específico.

Una forma de medir las expresiones faciales es a través de un proceso médico llamado electromiografía facial, el cual consiste en un sistema que hace uso de dos electrodos que se adhieren a la piel, para detectar y amplificar los pequeños impulsos eléctricos generados por los músculos faciales que rodean la ceja, los pómulos y la boca. Esta técnica no es invasiva y es capaz de brindar resultados precisos. Si bien no es invasiva, sí que es intrusiva, porque requiere del uso de electrodos y cables, arriesgando a que la incomodidad que pueda causar impacte en los resultados.[7]

2.2 Estado del arte

En la búsqueda de información realizada se encontraron miles de documentos relacionados con la identificación de expresiones faciales, esta búsqueda se pudo acotar realizando consultas por diferentes implementaciones con diversos métodos de IA. Adentrándose en este ambiente se empiezan a mencionar temas de optimización por lo que llega al objetivo final de las implementaciones en sistemas embebidos donde los recursos son limitados y el tema de optimización por *software* toma mayor impacto.

Ecuación de búsqueda:

((neural networks OR Artificial inteligence) AND (Computer vision OR facial recognition) AND (Deep Learning OR trained models) AND (process optimization OR resourses eficiency))

2.2.1 Real-Time Emotion Recognition from Facial Images using Raspberry Pi II

En la investigación se encontró una implementación que logró la identificación de expresiones en una Raspberry pi II, utilizaron el método *Haar cascade* y lograron identificar cinco emociones frontales con un 94% de precisión en sus medidas utilizando el *dataset CMU Multi PIE*. Se encontró una comparación de efectividad con otros modelos implementados en la literatura, se puede ver la comparativa en la tabla 1 donde se evidencia el porcentaje de efectividad en la predicción, la base de datos que se usó, el número de puntos que se identificaron del rostro, el procesador en el cual fue ejecutada la tarea y la técnica usada para la predicción de la expresión facial en tiempo real. [5]

En la tabla 1 se pueden visualizar los resultados de este trabajo con altos valores de precisión en el funcionamiento debido a que las pruebas que se realizaron fueron con 25 ejemplares de las bases de datos.

Tabla 1. Comparación resultados en diferentes sistemas embebidos [5]

[Author year]	Technique used	Dataset used	*Accuracy*	Number of points	*Time taken*
[Peng 2010]	*Canny filtter AAM Least Square method*	*JAFFE*	*85%*	*24*	-
[F. Abdat]	*Shi & Thomasi method, RBF SVM*	Cohnkanade *FEEDTUM*	*95%*	*38*	*721 ms with PC Intel Pentium 3,4 GHz*
[Rohit 2013]	*Local binary Patterns. SVM, Ada boost*	*JAFFE*	*86,67%*	*Texture Based*	227 ms (SVM) 1051 ms (adaboost) with Pc intel i3 2.2 GHz
[Myungho 2014]	*ASM, SVM classifier*	Cohn kanade	*72%*	*77*	*421,6 ms*
[Kamlesh 2014]	*AAM, LBP, Neural Network classifier*	*Cohn Kanade*	*88%*	*68*	-
Proposed Method	*ASM, Ada boost*	*CMU Multi PIE*	*94%*	*26*	*120 ms with raspberry Pi II (Linux ARM1 I7 6JZF, 900MHz)*

2.2.2 Deep Convolution Neural Network Implementation for Emotion Recognition System

Continuando con la investigación de la literatura este documento propone utilizar redes neuronales para lograr identificar el tipo de expresión facial, experimentalmente compararon dos tipos de *Deep networks* y llegaron a la conclusión de que la red convolucional tuvo un mejor rendimiento que una red neuronal profunda, primeramente, utilizaron detección facial para recortar solamente la imagen de la cara, extraerla del fondo y de esta manera tomar la información de la expresión facial y proceder a procesarla por una red neuronal para poder obtener una predicción acertada utilizando data base CK+. [2]

En las figuras 2 y 3 se puede observar el diseño y funcionamiento del trabajo descrito anteriormente.

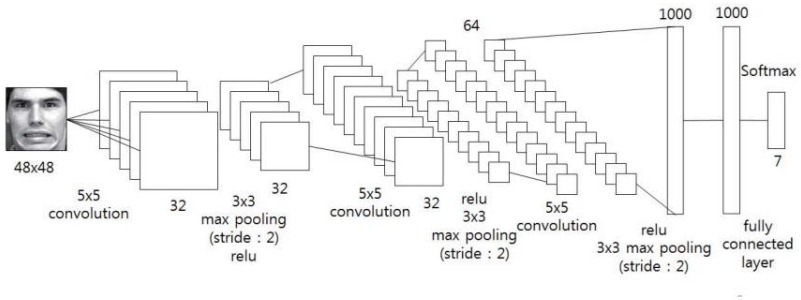

Figura 2. Arquitectura red neuronal utilizada [2]

Figura 3. Ejemplo de funcionamiento [2]

2.2.3 Deep Spiking Neural Network for Video-Based Disguise Face Recognition 0Based on Dynamic Facial Movements

En otro documento encontrado en la literatura [1] los autores proponen que, mediante el crecimiento exponencial de las redes sociales y los dispositivos inteligentes, la cara es uno de las llaves o contraseñas más poderosas para el reconocimiento de la identidad de una persona, se basaron principalmente en usar un *framework* llamado *VDFR Method* el cual está basado en modelos neuronales "*motion-sensitive*". La arquitectura final de este desarrollo consta de una realimentación jerarquizada SNN que consta de cinco capas.

- Capa de extracción de movimientos dinámicos: extrae diferentes vectores del video de entrada.
- Capa de extracción de características de alto nivel: se extraen características notorias de la imagen.
- Capa de codificación de picos: transforma vectores de características analógicas en una secuencia continua de patrones de picos.
- Capa de aprendizaje de patrones de picos: Con la codificación de la capa anterior se realiza un aprendizaje en esta capa.

- Capa de salida: Resultados.

2.2.4 *Facial expression recognition based on Gabor feature and neural network*

Continuando con la revisión a la literatura, se encontró un artículo [9] donde se realizó un reconocimiento de expresiones faciales utilizando *Gabor features* entre ellas la transformada de Wavelet, en primera instancia se explicó la estructura propuesta para el sistema, esta constaba de tres bloques principales

- *Face detection and positioning*: Este primer paso es donde se detecta el rostro (se hace uso de la librería DLIB)
- *Facial expression feature extraction*: En este bloque ya extraen las características para poder tener entrenada la red, se usa un 70% de la base de datos para entrenar y un 30% para testear
- *Facial expression clasification:* Este es el último bloque donde está el producto final

En la figura 4 se puede observar el flujo del proceso a mayor detalle, donde tanto la rama de entrenamiento como la rama de pruebas pasan por el mismo proceso a excepción que en la rama de entrenamiento se construyen los *Gabo filters*.

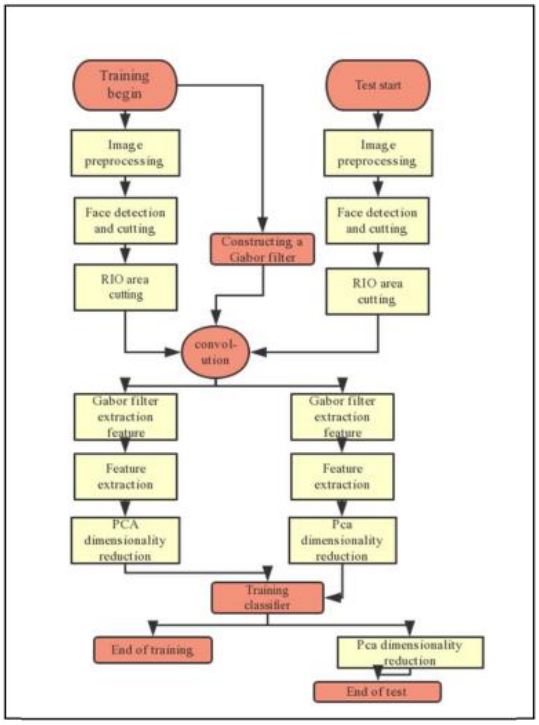

Figura 4. Modelo del sistema tomado de [9]

Con el modelo de reconocimiento facial propiedad de DLIB se pueden obtener nuestras ROI (*Region Of Interest*), que en este caso son dos imágenes diferentes, una donde se extraen los ojos y las cejas y otra donde se extrae la nariz y la boca como podemos observar en las siguientes imágenes (figura 5).

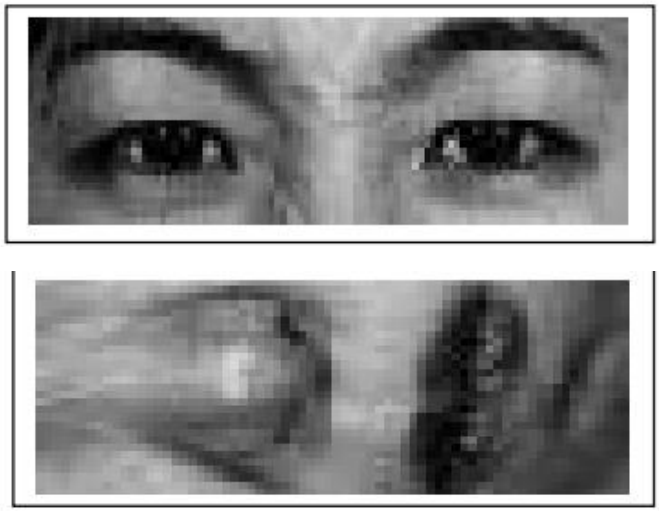

Figura 5. Imágenes extraídas para el entrenamiento de la red neuronal [9]

Por otro lado, hablando del *Gabor filter* es la posibilidad de describir diferentes características de distintos campos que un sistema no podría detectar fácilmente con un entrenamiento robusto. Se realizaron 40 Gabor filters que fueron implementados en Python donde se usaron los siguientes parámetros:

- Cinco diferentes escalas (λ=3,6,9,12,15)
- Ocho diferentes direcciones (θ=0, π/8, 3 π/8, π/2, 5 π/8, 3 π/4, 7 π/8)

La *Gabor Wavelet extraction feature* es una herramienta muy potente para el procesamiento de imágenes, pero finalmente como toda herramienta de la misma forma que nos brinda ventajas, también tiene debilidades o se complejiza en otros aspectos, en este caso el problema de esta extracción es que al pasar por los *Gabor filters* al ser 40 filtros la dimensión original de la imagen se multiplica por este número, finalmente tenemos una imagen 40 veces más grande que la original por lo que ahora el problema es tratar de disminuir dicha dimensión, la primera opción propuesta por el autor es una llamada *pooling* que es muy usada en *deep learning* para reducir dimensionalidad, este método trata de agrupar N pixeles de la misma área de la imagen y escoger el de mayor valor, y de esta manera reducir la dimensión de la imagen sin perder la esencia de esta. La segunda opción propuesta por el autor trata de un método llamada PCA, es un método estadístico que trata de reducción de dimensiones, el fundamento de este método es minimizar la perdida de información de la imagen mediante una compresión de los datos.

Continuando con el desarrollo de la solución final siguen con la clasificación de expresiones la cual lo hacen mediante una red neuronal BP, esta es una red *multi-layer* con el algoritmo de aprendizaje Widrow_hoffi en figura 6 se puede observar la estructura de la red neuronal utilizada.

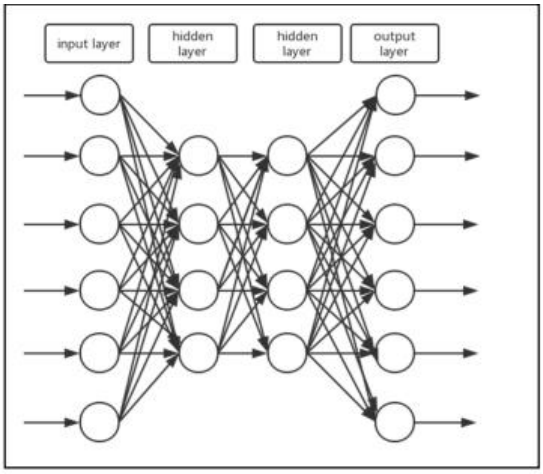

Figura 6. Arquitectura red neuronal [9]

Los parámetros para el entrenamiento de esta red fueron los siguientes:

Tabla 2. Parámetros de configuración para la red tomado de [9]

Parámetro	Valor
Learning_rate_base	0,0023
Learning_rate_decay	0,99
Regularization_rate	0,0001
Training_Strps	3000
Moving_aveage_decay	0,99

Con estos parámetros, después de realizar el entrenamiento con su *dataset* se obtuvieron los siguientes resultados

Tabla 3. iteraciones vs accuracy rate tomado de [9]

Dataset	*Accuracy rate*
> 0 iteraciones	0,093333
Después de 1000 iteraciones	0,85
Después de 2000 iteraciones	0,99
Después de 3000 iteraciones	0,8907

Finalmente se obtiene un resultado exitoso teniendo una tasa de reconocimiento promedio del 89%, superando otros métodos que también utilizaron el *Gabor filter*, esto traería como conclusión que al implementar este filtro junto con una red neuronal saca a relucir su gran potencial.

El principal error que se ve sobre este proyecto es el no tener en cuenta el factor de sobre entreno de la red neuronal, analizando la tabla 3 se puede observar que después de las 2000 iteraciones tiene un mayor *Accuracy rate* (tasa de precisión), aunque tener un 0,99 es algo muy complicado y puede tender a que sea inestable. Este tema no fue abarcado en su estudio y no dieron un argumento válido para seguir entrenando dicha red y como consecuencia disminuir su *Accuracy rate*.

Capítulo 3

3.1 Diseño de la solución

3.1.1 Algoritmos de reconocimiento facial

Para el diseño de la solución se tuvieron en cuenta dos algoritmos de detección facial (DLIB y haar cascade), haar cascade tiene un menor procesamiento que DLIB, pero esto se traduce en que es difícil conseguir una predicción acertada con la información que este brinda.

3.1.1.1 DLIB

El algoritmo de detección facial de DLIB, es un modelo pre entrenado que retorna 64 puntos esenciales de la cara dando coordenadas X, Y tal y como lo muestra la figura 7.

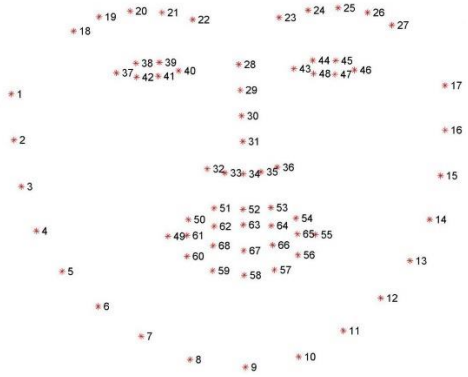

Figura 7. Coordenadas puntos x,y reconocimiento facial DLIB

Teniendo en cuenta los puntos retornados se puede observar que los que nos dan la información importante sobre una expresión facial son principalmente las cejas, ojos, nariz y boca, por lo tanto, los puntos del contorno facial se pueden ignorar para esta aplicación.+

3.1.1.2 Haar Cascade

El algoritmo de detección facial de *Haar Cascade* es una excelente opción para trabajar con pocos recursos, este es un modelo pre entrenado que retorna un recuadro enmarcando la cara reconocida como se puede observar en la figura 8, este algoritmo se usa en conteos de caras en fotografías.

Aun así, al ser un algoritmo que parece encajar bien en la solución, el problema radica en que no sólo se debe reconocer un rostro, el objetivo es reconocer expresiones faciales, por lo tanto, las coordenadas del recuadro que devuelve este modelo hace que el siguiente bloque de la solución se complique, esto debido a que no tenemos información valiosa que podamos procesar sin hacer uso de algoritmos de IA o ML avanzados, teniendo en cuenta que se trabaja con recursos limitados el algoritmo de *Haar Cascade* fue descartado.

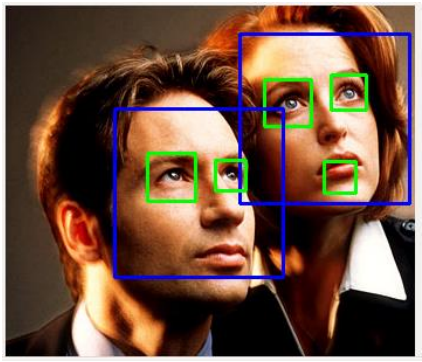

Figura 8. Ejemplo reconocimiento facial HaarCascade

3.1.2 Propuestas de solución algoritmos *Machine Learning*

3.1.2.1 KNN

El algoritmo de k vecinos más cercanos, es un clasificador que utiliza la proximidad para realizar predicciones sobre un conjunto de datos, hay que asignar etiquetas a la base de datos sobre un voto mayoritario y para realizar una predicción sobre una clasificación se toma el promedio de los vecinos más cercanos, dándonos así el valor discreto más cercano a la predicción como se observa en la figura 9 donde están los datos de clase A, clase B, las diferentes constantes K y el resultado deseado.

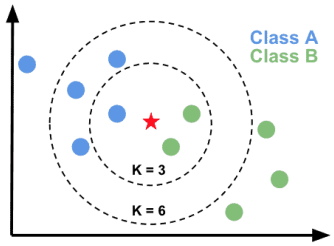

Figura 9. Algoritmo clasificación KNN (tomado de [15]).

3.1.2.2 SVM

Support Vector Machine es un algoritmo de *machine learning* supervisado que usa algoritmos de clasificación, siempre y cuando se le dé una base de datos con qué entrenar, el algoritmo estará listo para categorizar los parámetros ingresados al sistema.

Lo que se trata de hacer entender con la figura 10 es que los parámetros varían hasta encontrar el mejor hiperplano, de esta manera todos los datos ingresados que se encuentren encima del plano se discretizarán de un valor y todos los que se ingresen debajo del plano tendrán otro valor discretizado previamente habiendo etiquetado la base de datos.

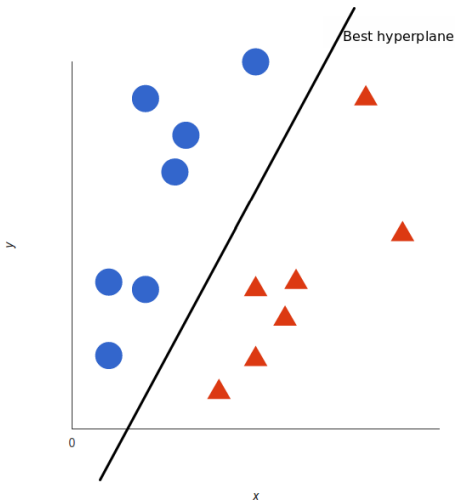

Figura 10. Algoritmo clasificación SVM [15].

3.1.2.3 *Random Forest* (bosque aleatorio)

Algoritmo de clasificación que predice acciones futuras de un usuario por medio de árboles de decisión, bases de datos etiquetadas y valores de entrada podrá dar un estimado sobre la etiqueta del dato entrante como se puede ver en la figura 11, ya que el sistema realiza análisis sobre los datos de entrenamiento y sobre estos resultados estima las próximas entradas al sistema.

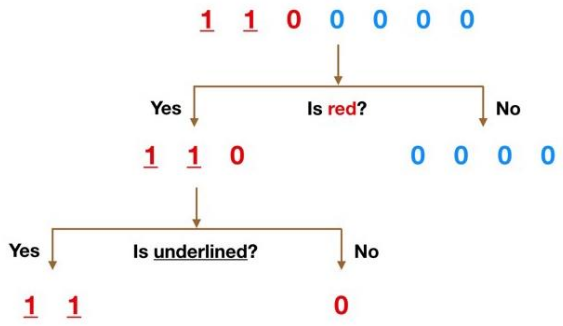

Figura 11. Algoritmo random forest [15].

3.1.3 Solución planteada

Para la solución planteada se elige el algoritmo de detección facial *DLIB*, aunque *Haar Cascade* sea un algoritmo que exige menos trabajo al sistema embebido al intentar acoplarlo con un algoritmo de *machine learning*, este algoritmo daba muy poca información con la cual podíamos realiza una predicción correcta, debido a que sólo nos daba las coordenadas del marco de la cara detectada, información la cual no era de mucha utilidad para la implementación de un algortimo de *Machine Learning* simple.

La estrategia que se utilizó con el algortimo *DLIB* es que, al tener 64 puntos podemos tomar las distancias teniendo uno de referencia, en este caso se tomó el punto central de la nariz (punto #34), los puntos del contorno facial no brindan mucha información para el sistema, por lo tanto, se ignoran (puntos 1-17), este proceso se hace con cada imagen para el entrenamiento del algoritmo de *Machine Learning* con estos datos se procede a realizar el entrenamiento, en el planteamiento de la solución no se eligió como tal un algoritmo de *Machine Learning*, esta decisión se tomó con los resultados y rendimiento obtenido con las pruebas funcionales.

En la figura 12 se observa el diagrama de bloques del sistema de la solución propuesta.

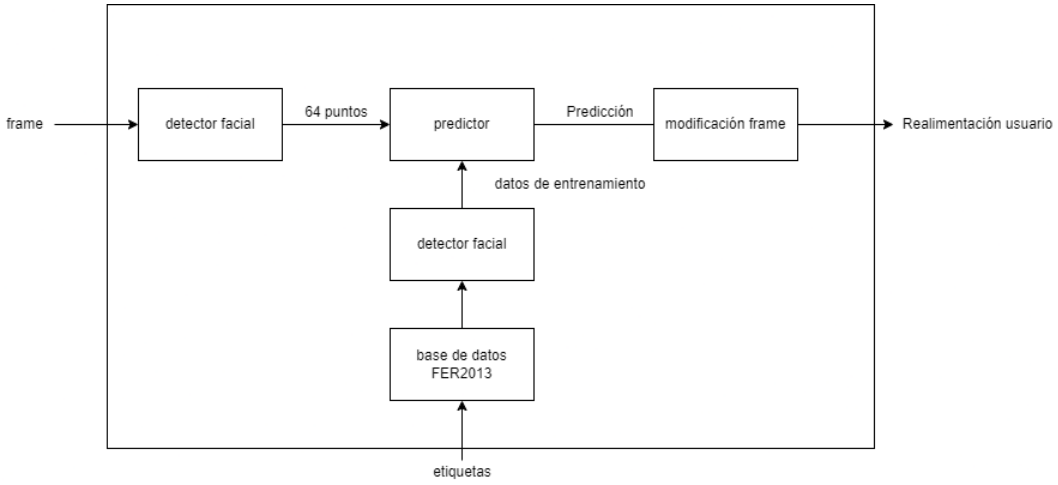

Figura 12. Solución propuesta (creación propia).

La base de datos que se definió para el entrenamiento y pruebas de este sistema es FER2013 la cual es de frecuente uso en los proyecto de *machine learning* para reconocimiento de expresiones faciales como por ejemplo [2] y [5], esta se eligió debido a que es una base de datos diversa con diferentes fotografías que contienen personas con diferentes rasgos, características y posiciones, esto con el fin de exponer el sistema a casos críticos y cuando sean las condiciones ideales funcione correctamente sin impedimentos, en la tabla 4 se pueden observar el número de muestras de esta base de datos seccionado por género y emociones, por otro lado en la tabla 5 se puede observar las muestras de perfil de la base de datos.

Tabla 4. Numero de muestras seccionado por género y emociones FER2013

	Hombre	Mujer	Total
Feliz	4113	3102	7215
Triste	2657	2174	4830
Sorprendido	1712	1459	3171
Total	8482	6735	15216

Tabla 5. Numero de muestras de perfil seccionado por género y emociones FER2013

	Hombre	Mujer	Total
Feliz	708	534	1242
Triste	380	299	679
Sorprendido	267	226	492
Total	1355	1059	2413

Las expresiones faciales para reconocer que se escogieron fueron: sorpresa, tristeza y felicidad, esto debido a sus grandes diferencias en la visualización del rostro debido a que otra emoción que era candidata para implementarse era disgusto y neutral, pero debido a la similitud que se tenía con la emoción de tristeza estas se descartaron, se optó por la emoción de sorpresa debido a que esta tercera emoción tiene unos gestos característicos como lo son la boca abierta o los ojos muy abiertos, de esta manera el sistema tendrá manera de poder realizar una mejor clasificación.

3.1.4 Multi reconocimiento de expresiones faciales

Para satisfacer el requerimiento de reconocer expresiones faciales en al menos dos rostros, la estrategia que se planteó se basa en que a partir de reconocer el primer rostro se toman las coordenadas de las esquinas del reconocimiento, después de realizar la predicción se rellena con ceros esta área de la imagen proceso el cual oculta el rostro, por lo que esta misma imagen modificada vuelve a ingresar al sistema, detecta el siguiente rostro, realiza la predicción y el proceso correspondiente, esto se vuelve a repetir hasta que no encuentre otro rostro.

Como ejemplo de reconocimiento de dos expresiones faciales, se tiene inicialmente la imagen original (figura 13).

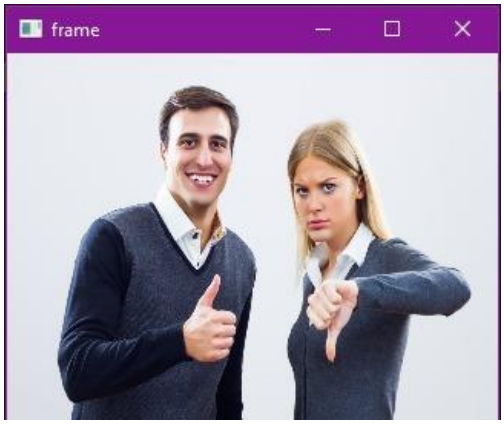

Figura 13. Imagen inicial dos personas tomado de freepik.es

Al pasar por el primer reconocimiento, se realiza todo el proceso con el primer rostro, posteriormente se anula esta región de la imagen estableciendo el valor de los pixeles en cero, lo que traduce visualmente a que esta zona de la imagen se verá en negro (figura 14).

Figura 14. Imagen después del primer reconocimiento tomado de freepik.es

Posteriormente se realiza el mismo proceso con el segundo rostro (figura 15), en caso de que no encuentre otro rostro en la imagen el algoritmo no tendrá ningún inconveniente en su ejecución.

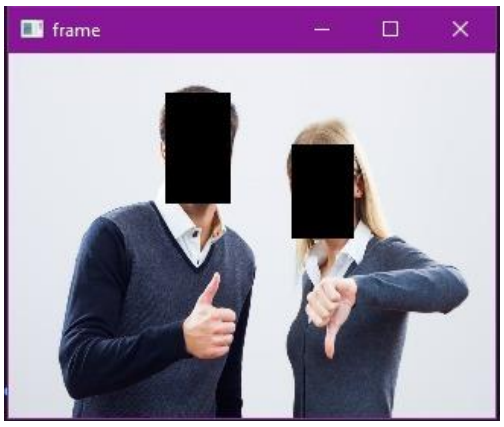

Figura 15. Imagen resultante del segundo reconocimiento tomado de freepik.es

Las imágenes mostradas en la figura 14 y figura 15 son una copia de la imagen original para poder realizar este multi reconocimiento, finalmente, la imagen que se le va a retornar al usuario mostrando las expresiones faciales reconocidas es una copia diferente de la imagen original donde está destinada a mostrar la interfaz de usuario diseñada para el sistema (figura 16) la cual consta

de una realimentación de las emociones detectadas mediante emojis en la parte superior del rostro y en la parte inferior derecha la imagen original.

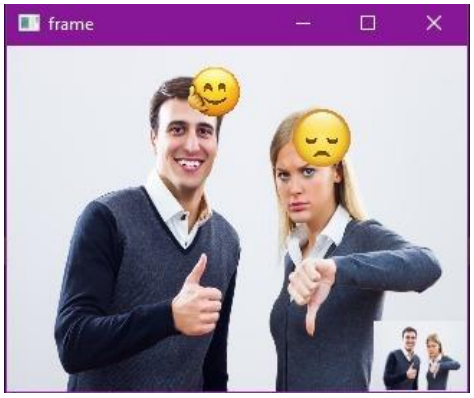

Figura 16. Realimentación al usuario tomado de freepik.es

Como ejemplo de reconocimiento de tres expresiones se tiene inicialmente la imagen original (figura 17) donde se puede observar a tres mujeres.

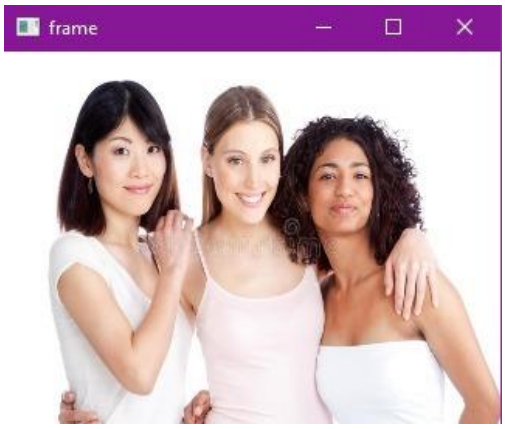

Figura 17. Imagen original 3 rostros tomado de freepik.es

Se realiza el mismo procedimiento que se realizó para el reconocimiento de los dos rostros, (figura 18,19,20), en la figura 18 se puede observar como reconoció el primer rostro y anulo esta región de la imagen poniendo el valor de sus pixeles en cero.

Figura 18. Primer reconocimiento tomado de freepik.es

En la figura 19 podemos observar como reconoció el segundo rostro y anulo esta región de la imagen estableciendo el valor de sus pixeles en cero

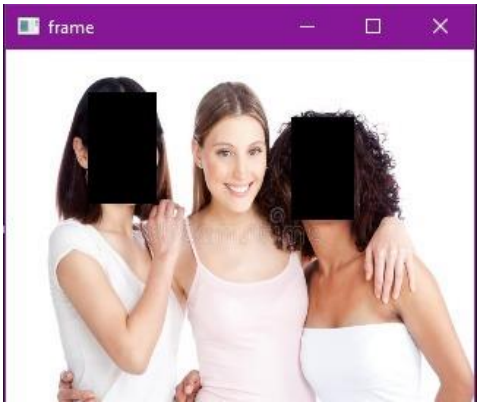

Figura 19. Segundo reconocimiento tomado de freepik.es

En la figura 20 se observa como realiza el tercer reconocimiento y anula dicha región de la imagen estableciendo el valor de estos pixeles encero.

Figura 20 Tercer reconocimiento tomado de freepik.es

Finalmente se obtiene la realimentación al usuario mediante la interfaz diseñada (figura 21) donde se obtiene una predicción de que las tres mujeres se encuentran felices lo cual es correcto.

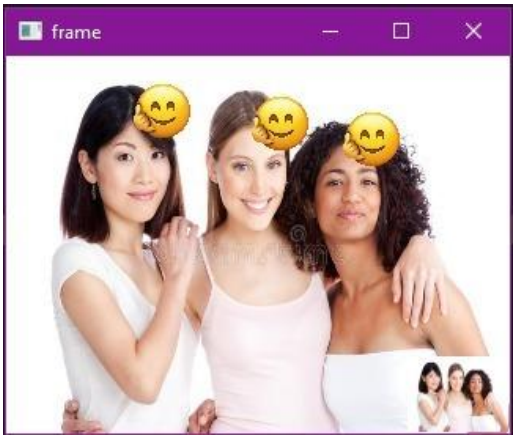

Figura 21. Realimentación al usuario tomado de freepik.es

Capítulo 4

4.1 Migración de la solución a los sistemas embebidos

Una vez se estructuró el código de tal manera que fuera funcional en un computador convencional, el siguiente paso a seguir fue migrar a los sistemas embebidos Raspberry pi 4 y Jetson Nano los cuales se desempeñan sobre sistema operativo *Linux* pero en diferentes extensiones, *Raspbian* y Ubuntu respectivamente. En esta sección del libro se hablará sobre los diferentes procesos que se aplicaron para poder migrar a las tarjetas de placa reducida.

4.2 Instalación de entornos de programación y plataformas de escritorio remoto:

En el proceso de migración fue necesario escoger entornos de programación en los cuales hubiese la posibilidad de compilar código en lenguaje *Python*, en el caso de la *Raspberry* el sistema operativo *Raspbian* por defecto trae la herramienta *thonny* que está bien optimizada, caso contrario al de la *NVIDIA Jetson nano* en la cual hubo que instalar a través de comandos en la terminal *Visual Studio Code* con el fin de contar con un entorno de programación, esto da una total ventaja de un sistema sobre otro en cuanto a la migración mientras que en uno todo está optimizado y ya viene con las herramientas necesarias, en el otro sistema hay que realizar una cantidad de pasos previos para poder empezar a compilar el código.

En cuanto a las plataformas de escritorio remoto se escogió la alternativa de VNC *server* para acceder a ambos sistemas embebidos a través de estas conexiones cifradas y trabajar con comodidad sin necesidad de conectar periféricos a estas sino con tal de que estuvieran conectadas a internet y energizadas se podía acceder por escritorio remoto a través de un computador que estuviera conectado a la misma red. En el caso de Raspberry ya el sistema operativo lo traía instalado, pero en *NVIDIA Jetson Nano* fue necesario instalarlo a través de comandos en la terminal, contando con el problema que accediendo por conexión remota se visualizaba a una resolución 3:4, esto se pudo arreglar modificando unos archivos del sistema.

4.3 Instalación de Python y librerías:

Uno de los procesos que demandó más tiempo en la migración a los sistemas embebidos fue la instalación de Python y sus componentes necesarios para realizar los diferentes procesos de reconocimiento facial, inteligencia artificial, procesamiento de imágenes, lectura de archivos del sistema y demás características entre ellas las siguientes:

- DLIB: Librería hecha en C++ la cual contiene algoritmos de *Machine Learning*, de la cual se utiliza la estancia de detección facial.
- *Opencv*: Librería de procesamiento de imágenes la cual se utiliza para modificar la imagen en vivo, realizar la interfaz del usuario, sobreponer los emoticonos de realimentación y crear *labels* para informar los *frames* por segundo del video en vivo.
- *Cmake*: complemento necesario para tener la posibilidad de instalar DLIB en cualquier sistema operativo.
- *Sklearn*: librería para realizar los análisis de predicción e inteligencia artificial.
- *Cvzone*: librería de visión por computadora.

Al instalar las librerías y complementos en sistema operativo *Linux* nos encontramos con varias diferencias a Windows, por ejemplo, para Raspberry se descubrió que hay una versión de *opencv*

que está exclusivamente diseñada y optimizada para correr en este sistema y si se trata de realizar la instalación normal dará error.

Se encontró la novedad en la tarjeta *NVIDIA Jetson Nano* que las librerías se demoraban una cantidad de tiempo fuera de lo normal ya que en la *Raspberry* o en un computador de escritorio solía ser un proceso relativamente corto, a diferencia de esto podrían llegar a tardar hasta cuatro horas en la instalación de las librerías más complejas como DLIB o *Opencv*, profundizando en los foros de la comunidad no es un incidente que solo haya pasado en esta ocasión, sino que está hecho de esta forma para que dentro de la instalación se optimice al máximo la compilación y el uso de la GPU con la que cuenta la tarjeta, de esta manera se aprovecha al máximo el *hardware* para obtener mejores resultados a la hora de realizar procesamiento de imágenes contando con una tarjeta gráfica ya que este tipo de componentes están diseñados para llevar a cabo este tipo de tareas.

Al instalar las librerías en la *NVIDIA Jetson Nano* fue necesario tener una versión de *Python* específica para instalar DLIB e instalar los complementos *lingtk2.0.dev y pkg-config* para que el complemento *Cmake* y sus configuraciones funcionaran correctamente, una vez realizado este proceso y reinstalar *opencv* el sistema funcionó correctamente.

4.4 Integración de cámara USB con *Linux* (*Raspbian*):

Adaptando la *Raspberry PI* a las funcionalidades del código se evidenció que había un problema a la hora de identificar la cámara USB en este sistema operativo, hay un comando especifico de la librería *opencv* para acceder a las imágenes de la cámara predetermina del sistema el cual es el siguiente: “cv2.VideoCapture(0)” pero se encontró que en el momento de ejecutar este comando en la migración a Raspbian el interpretador de *Python* informaba el siguiente error:

VIDEOIO ERROR: V4L: can't open camera by index 0

Al correr el comando lsusb (figura 22) en la terminal de *Linux* se puede observar que el sistema está reconociendo la cámara como Microdia con ID 0c45:636b.

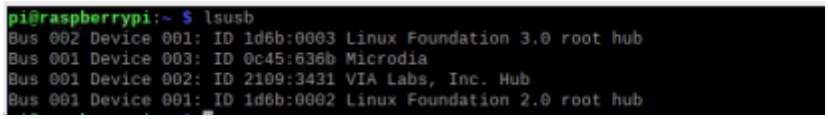

Figura 22. resultado comando lsusb

En la figura 23 se observa el comando para obtener información sobre la cámara:

Figura 23. resultado comando sudo dmesg | grep -i camera

Al parecer es común al intentar obtener imagen de la cámara con la librería cv2 y tener como respuesta el error de que no se encuentre la cámara o no se pueda abrir, al investigar diferentes foros de internet donde los programadores expresaban tener el mismo error recomendaban ejecutar el comando “sudo modprobe bcm2835-v4l2” en la terminal del equipo y hacer un *reboot*, acción que tuvo un resultado positivo e hizo que el comando pudiese acceder a la imagen en vivo de la cámara usb conectada al sistema.

Resultados de la migración:

La visualización de imágenes de la cámara predeterminada del sistema y la identificación facial es exitosa con la novedad de que brinda muy pocos FPS (bajo rendimiento y fluidez) y se evidencia un *delay* (retraso) de dos segundos entre lo que sucede en tiempo real y la imagen mostrada en pantalla.

4.5 Comparación y validación de funcionamiento en todos los sistemas (*PC, Jetson Nano, Raspberry py y eMotion*):

En la figura 24, 25 y 26 se puede observar el funcionamiento de las emociones "Feliz, Triste y Sorprendido" en cada uno de los sistemas. En cuanto al resultado de la emoción determinada, todos los sistemas coinciden, el cambio que se nota en la fluidez y latencia del video debido a la diferencia de *hardware* en cada uno de ellos, ya que el PC y la *Jetson Nano* cuentan con tarjeta gráfica para procesar las imágenes.

El software *eMotion* detecta las emociones y las clasifica en un porcentaje que brinda la certeza con la que el sistema asegura que corresponde a la emoción detectada en tiempo real, con las emociones "Feliz" y "Sorprendido" en condiciones ideales como buena luz, rostro frente a la cámara, buen enfoque y demás; brinda un 100% de certeza a diferencia de la emoción triste que afirma que es un 90% triste 6% enojado 2% sorprendido y 1% neutro, con esta información se puede inferir que el sistema *eMotion* puede llegar a variar su predicción cuando se trata de la emoción triste debido a su gran similitud en la gesticulación del rostro con las demás emociones, esta variación tiene sentido y al compararlo con el aplicativo demostrado en este documento, de igual manera el resultado sería la emoción con más porcentaje de certeza.

La comparación y validación de funcionamiento se realiza con el *software* mencionado anteriormente, debido a que es un ejemplar que cumple con todas las características y criterios relacionadas con el objetivo funcional de este trabajo de grado, aunque entrega los resultados sea con un porcentaje de certeza y no por estados de emoción como lo realiza el aplicativo diseñado, de igual manera se puede tomar como una referencia confiable para validar el funcionamiento.

Figura 24. Funcionamiento emoción "feliz" en todos los sistemas.

Figura 25. Funcionamiento emoción "triste" en todos los sistemas.

Figura 26. Funcionamiento emoción "sorprendido" en todos los sistemas.

Capítulo 5

5.1 Resultados

Para la evaluación de los resultados se utilizaron las siguientes métricas

- *Accuracy* (precisión): Con esta métrica se evalúa el porcentaje de aciertos del sistema respecto a la cantidad total de datos de prueba que le fue suministrado.
- Coeficiente de correlación de Matthew's: También abreviado como MCC, es una herramienta estadística para la evaluación de modelos, tiene como trabajo calibrar o medir la diferencia entre los valores pronosticados y los valores reales guiándose de la matriz de confusión.
- *F1 Score*: *Precision* y *Recall* son los dos componentes principales de esta métrica, el objetivo del F1 *Score* es combinar estos dos componentes en una sola métrica.

Para la evaluación de los sistemas se realizaron dos procesos diferentes, el primero que se llamará **Proceso A** el sistema fue evaluando con una base de datos de rostros los cuales están la mayoría en condiciones ideales para el reconocimiento, por otro lado, en el **Proceso B** el sistema fue evaluado con una base de datos más exigente donde se encuentran casos no tan ideales para el sistema lo cual lo trata de exigir al máximo para realizar un reconocimiento exitoso.

En la figura 27 se puede observar un ejemplo de una imagen de la base de datos usada para el proceso A, en la figura 28 se tiene un ejemplo de una imagen de la base de datos usada para el proceso B, se puede apreciar que la base de datos del proceso B, como se mencionó anteriormente, se tienen casos no muy ideales para el sistema.

Figura 27. Ejemplo base de datos condiciones ideales [14]

Figura 28. Ejemplo base de datos exigente [13].

5.1.1 Proceso A

Para la evaluación del sistema mediante este proceso se utilizó la base de datos ckplus, una base de datos muy famosa entre los proyectos de *Machine Learning* en expresiones faciales, debido a la gran cantidad de sujetos de prueba que tiene incluyendo diferentes características físicas, entre ellas color de piel o rasgos fisionómicos.

Como primer paso en este proceso de evaluación de métricas se realizó con el sistema identificando dos expresiones faciales (feliz y triste) y los resultados obtenidos son los siguientes:

En la figura 29 podemos observar una gran diferencia de rendimiento del algoritmo KNN con un 56.58% de precisión para hombres y un 46.44% para mujeres, en cambio los algoritmos de *SVM* y *Random Forest* retornan unos mejores resultados, teniendo una precisión del 89% para hombres y 94.5% para mujeres en *Random Forest* y una precisión del 93% para hombres y 81.86% para mujeres en SVM, esta diferencia entre géneros se puede dar debido a que los rostros femeninos tienen una composición diferente y el *Random Forest* puede realizar una mejor correlación para estas características que el SVM.

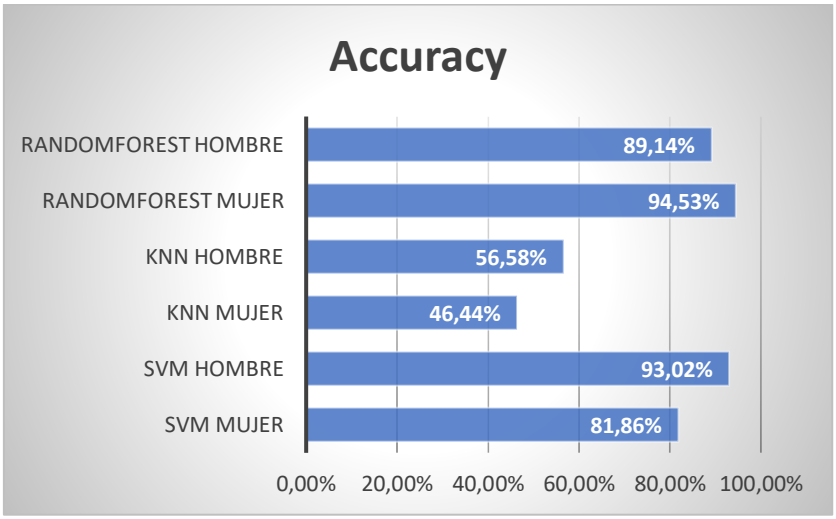

Figura 29. Métrica accuracy proceso A dos emociones.

En la figura 30 podemos observar los resultados del Coeficiente de correlación de Matthew's, hay que tener en cuenta que para esta métrica el resultado puede variar entre cero y uno donde cero es una predicción completamente aleatoria respecto a los datos reales y uno es el mejor resultado respecto a los datos evaluados y los datos reales.

Entre los tres algoritmos de *Machine Learning*, de nuevo, KNN obtiene unos resultados no deseados respecto a los otros dos algoritmos; en cuanto a los resultados de *Random Forest* se obtiene un coeficiente de 0.85 para hombres y 0.7831 para mujeres, finalmente para SVM se obtuvieron unos resultados de 0.86 para hombre y 0.669 para mujer.

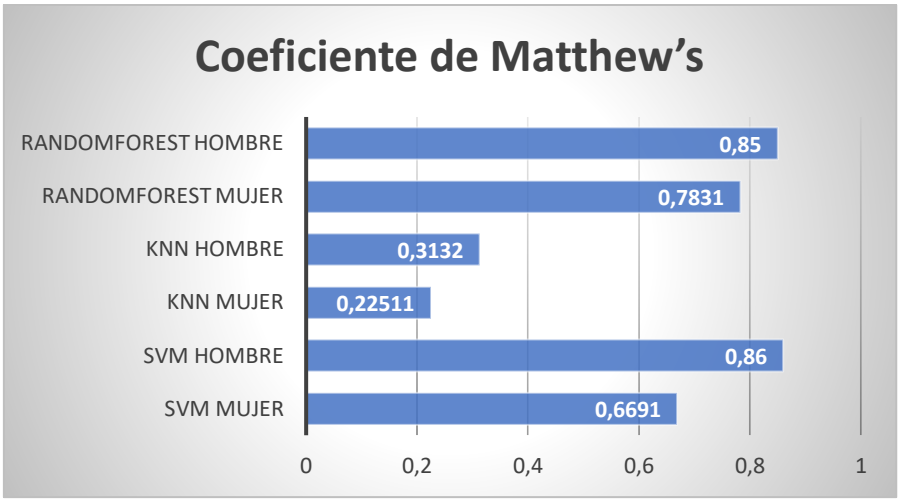

Figura 30. Métrica coeficiente de Matthews proceso A dos emociones.

En la figura 31 podemos observar los resultados del F1 *score*, para esta métrica también hay que tener en cuenta que su valor varió entre cero y uno, donde cero es un muy mal resultado donde el sistema es completamente aleatorio, y uno es un resultado perfecto, donde el sistema no tiene fallos.

Entre los tres algoritmos de *Machine Learning* esta vez no hubo una diferencia considerable entre los tres, aunque tomando como guía los resultados de la figura 29 y figura 30 se debe tomar una decisión entre elegir el algoritmo de SVM o *Random Forest*.

Algo adicional que se implementó en este proyecto fue agregar una tercera emoción la cual corresponde a sorpresa y se realiza la misma evaluación tal cual se realizó con dos emociones.

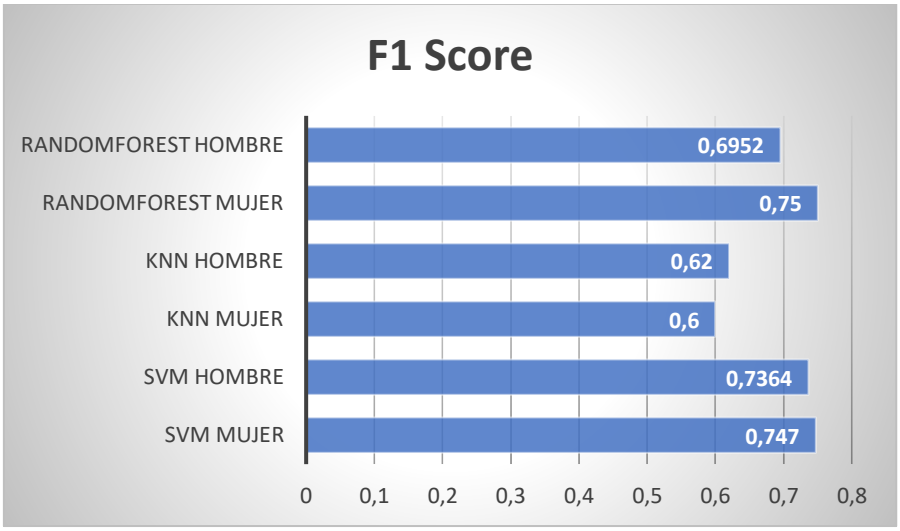

Figura 31. Métrica F1 score proceso A dos emociones.

En la figura 32 se puede ver la métrica de *accuracy* evaluada en los tres sistemas, aquí se puede observar que SVM tiene una superioridad frente a *Random Forest* y *KNN*.

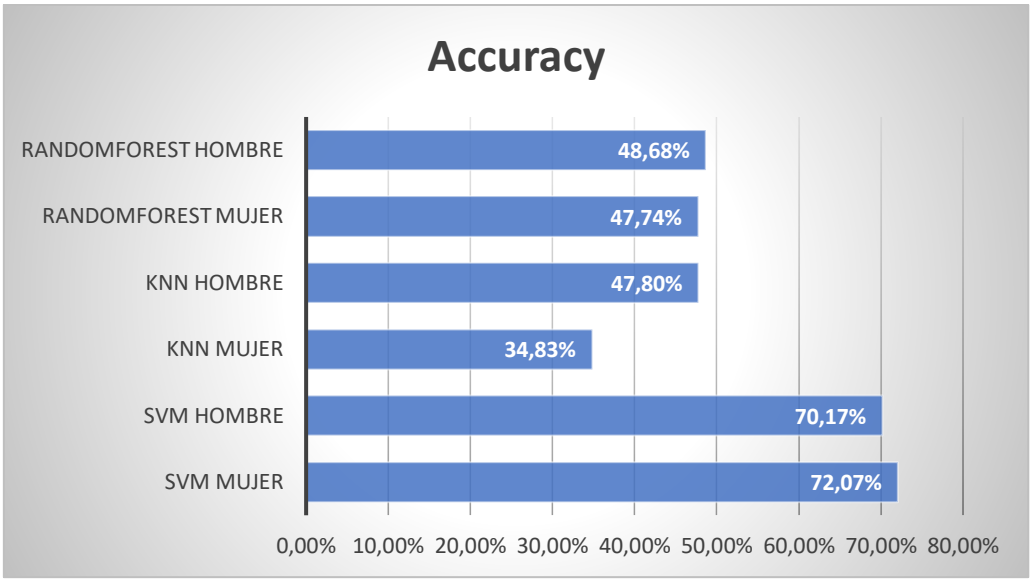

Figura 32. Accuracy proceso A tres emociones

En la figura 33 se puede observar el resultado del Coeficiente de *Matthew's*, en esta métrica también se puede apreciar que SVM tiene una superioridad notable frente a KNN y *Random Forest*.

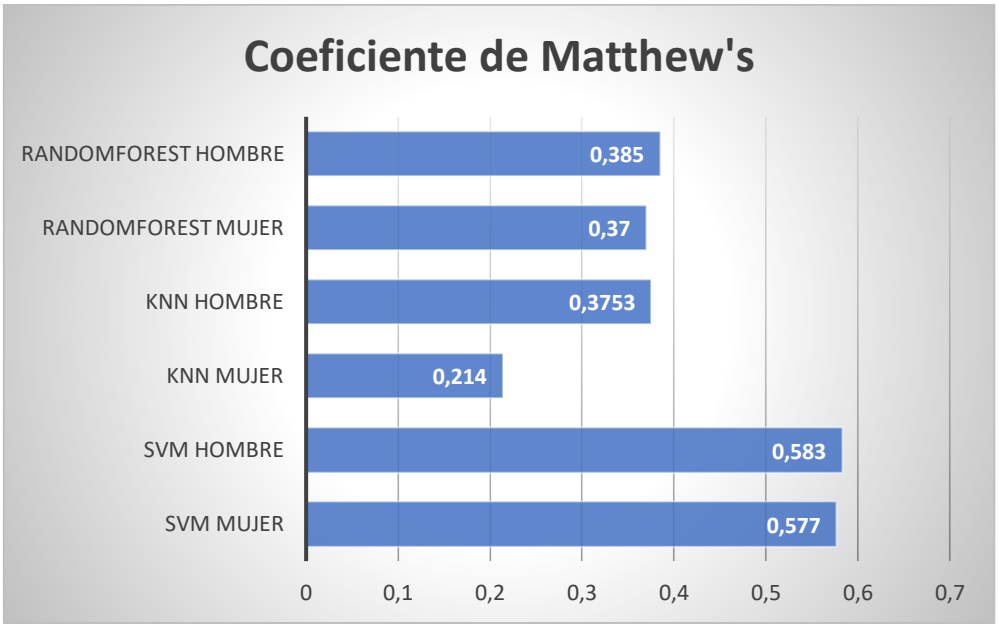

Figura 33. Coeficiente de Matthew's proceso A tres emociones

En la figura 34 se pueden observar los resultados de la métrica F1 Score en la cual nuevamente vemos una dominancia del algoritmo de SVM sobre *Random Forest* y KNN.

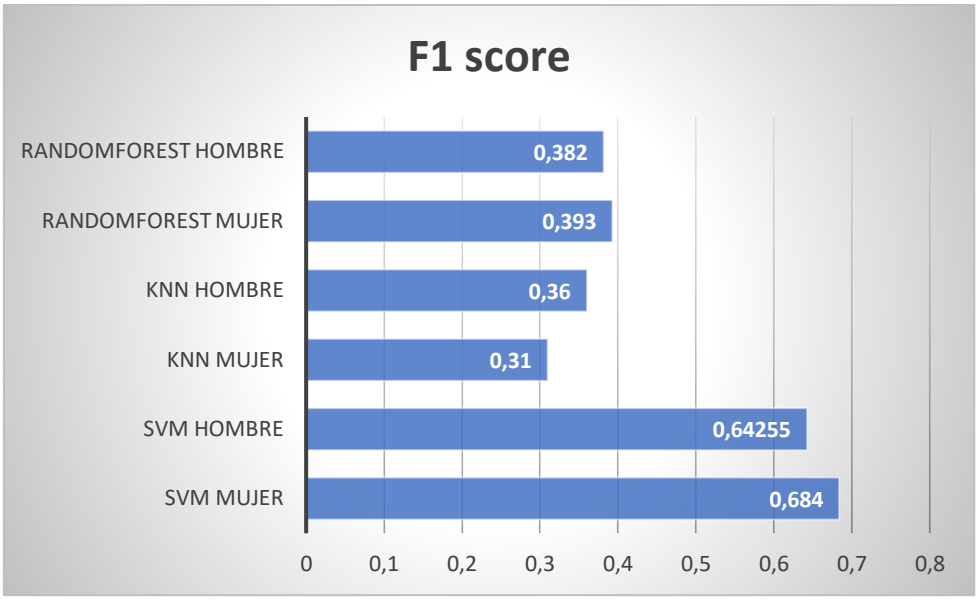

Figura 34. F1 score proceso A tres emociones

5.1.2 Proceso B

Para la evaluación del sistema mediante este proceso se utilizó la base de datos FER2013, una base de datos muy famosa entre los proyectos de *Machine Learning* en expresiones faciales, esta base de datos se caracteriza por tener rostros en diferentes posiciones lo cual lleva al sistema a casos extremos.

De igual manera que en el proceso A esta evaluación se realizó para dos expresiones faciales (feliz y triste)

En la figura 35 se pueden observar los resultados de *accuracy*, KNN está por debajo del rendimiento de los algoritmos de *Random Forest* y SVM donde estos tienen exactamente el mismo rendimiento.

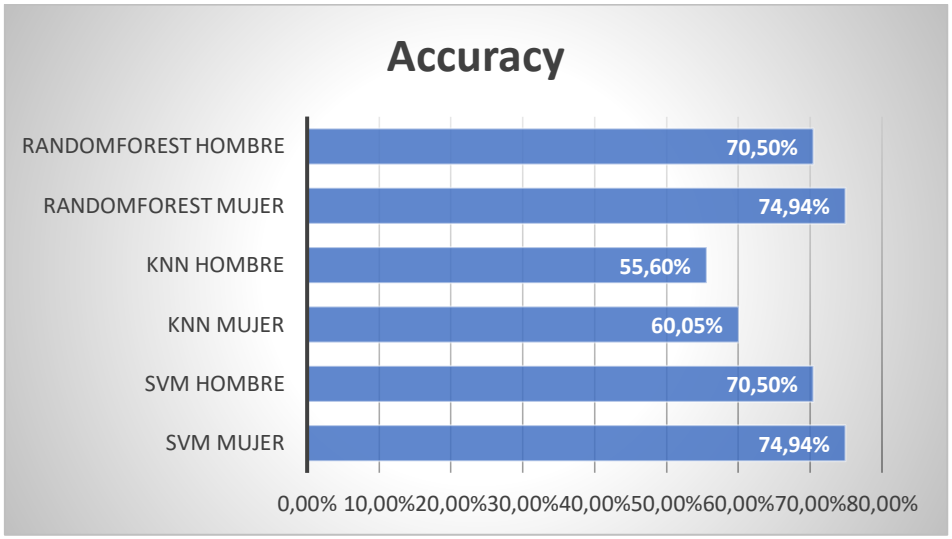

Figura 35. Accuracy proceso B dos emociones.

En la figura 36 se pueden observar los resultados del Coeficiente de Matthew's donde el algoritmo de KNN tiene un rendimiento muy inferior a los demás, nuevamente *Random Forest* y SVM tienen resultados similares.

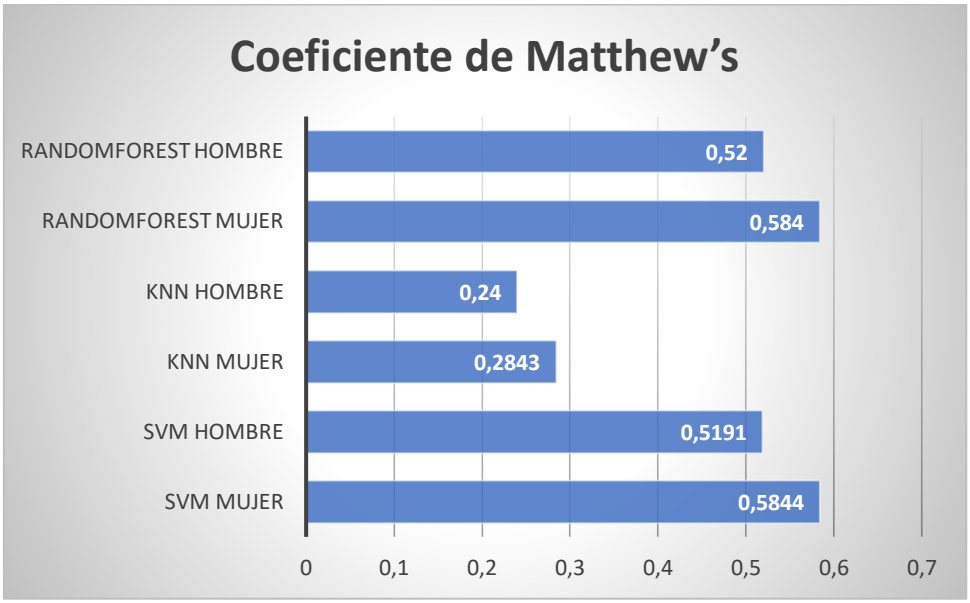

Figura 36. Coeficiente de Matthew's proceso B dos emociones.

En la figura 37 se pueden observar los resultados de la métrica *F1 Score* para los tres algoritmos de *Machine Learning*, donde se ve que SVM es el algoritmo más estable en estos resultados.

De igual manera que se realizó con el proceso A, en el proceso B se realizaron esta misma evaluación de métricas con el sistema de reconocimiento de tres emociones.

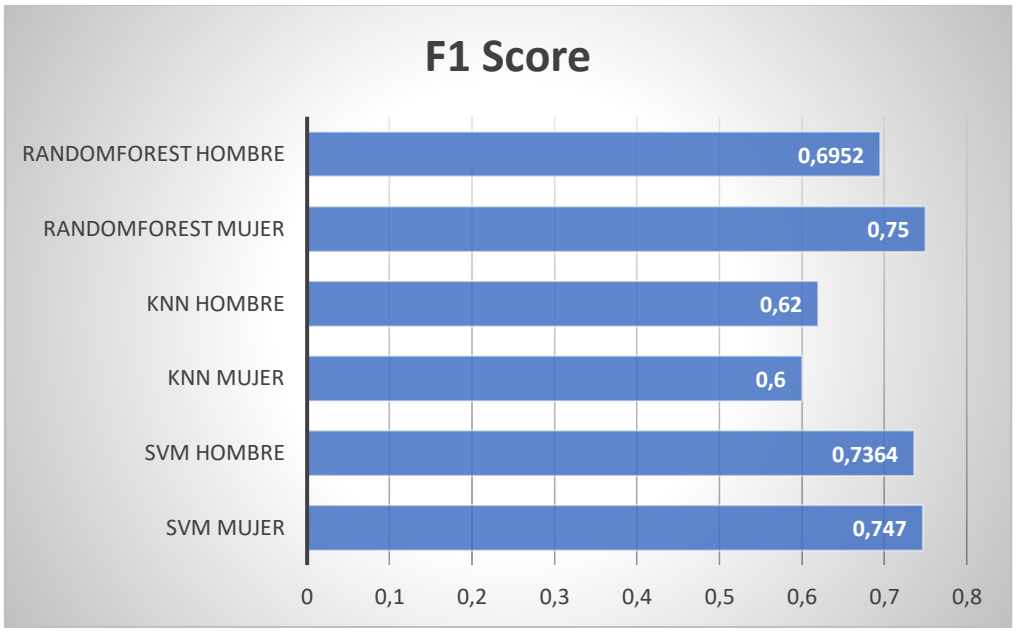

Figura 37. F1 score proceso B dos emociones.

En la figura 38 podemos observar como el algoritmo de SVM mantiene sus resultados estables con una perdida no muy significativa mientras que *Random Forest* y *KNN* no rinden de manera óptima para el reconocimiento de tres emociones.

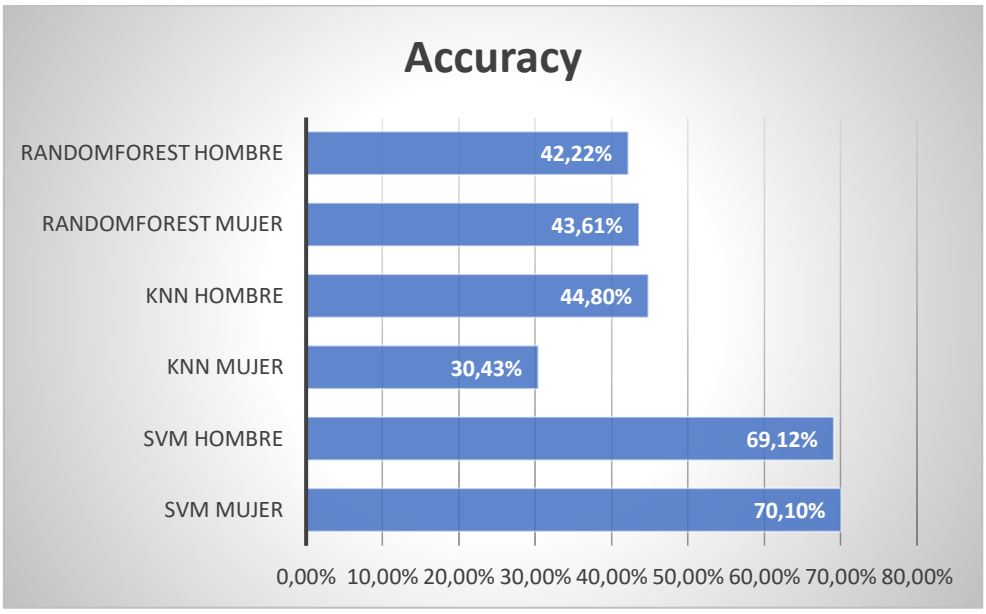

Figura 38. Accuracy proceso B tres emociones.

En la figura 39 se puede observar el resultado del coeficiente de *Matthew's*, esta es la métrica más influyente en la decisión sobre cuál algoritmo es el elegido entre los tres.

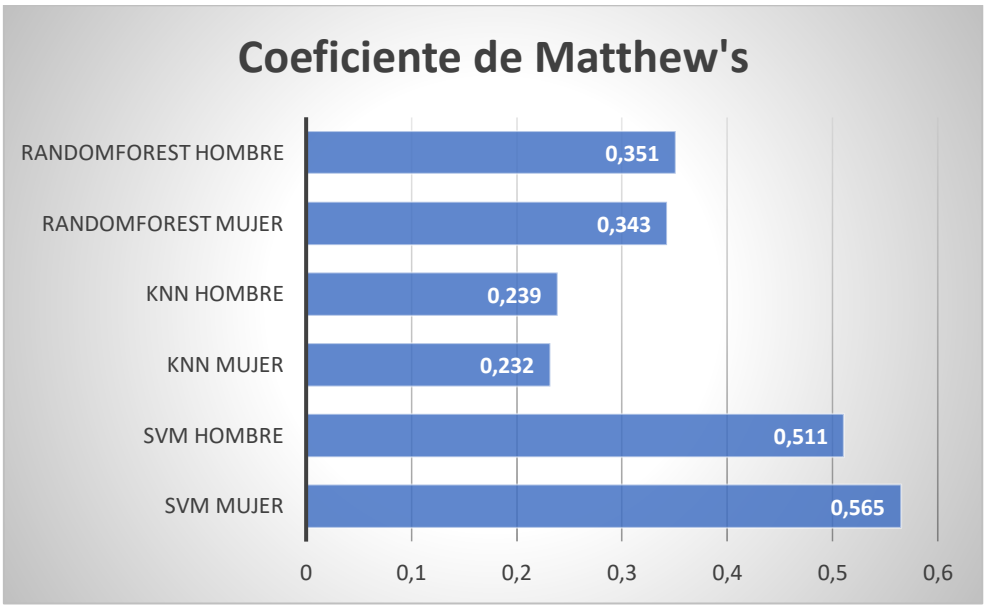

Figura 39. Coeficiente de Matthew's proceso B tres emociones.

Finalmente, en la figura 40 se puede observar el resultado de la métrica F1 *Score*, nuevamente, el algoritmo de SVM teniendo una superioridad notable frente a los demás algoritmos de *Machine Learning*.

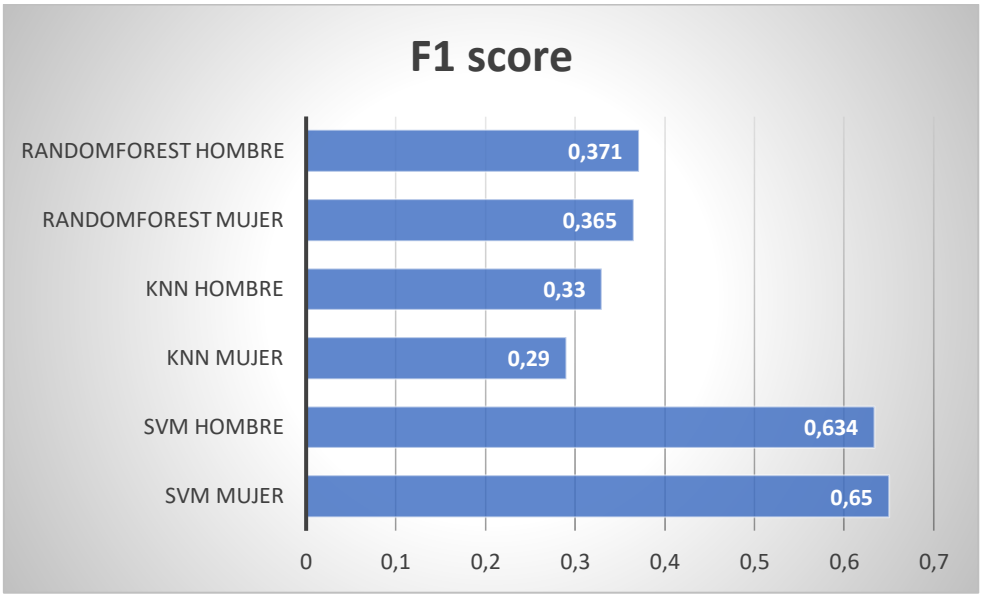

Figura 40. F1 Score Proceso B tres emociones.

5.2 Matriz de confusión

La matriz de confusión obtenida para el algoritmo SVM para tres emociones se muestra en la tabla 6, para el algoritmo KNN se muestra en la tabla 7 y para el algoritmo *Random Forest* se muestra en la tabla 8.

Tabla 6. Matriz de confusión SVM

	Feliz detectado	Triste detectado	Sorpresa detectado
Etiqueta feliz	88,12%	7,6%	4,22%
Etiqueta triste	17,41%	70,78%	11,79%
Etiqueta sorpresa	15,9%	11,59%	72,46%

Tabla 7. Matriz confusión KNN

	Feliz detectado	Triste detectado	Sorpresa detectado
Etiqueta feliz	84,49%	11,11%	5,39%
Etiqueta triste	50,33%	33,48%	16,17%
Etiqueta sorpresa	54,34%	7,74%	37,91%

Tabla 8. Matriz confusión Random Forest

	Feliz detectado	Triste detectado	Sorpresa detectado
Etiqueta feliz	97,30%	2,69%	0%
Etiqueta triste	21,91%	78,09%	0%
Etiqueta sorpresa	53,32%	46,68%	0%

5.2.1 Matriz de confusión personas de perfil

Las matrices de confusión teniendo en cuenta las muestras de perfil para los algoritmos SVM, KNN y Random Forest se muestran en las tablas 9, 10 y 11 respectivamente.

Tabla 9. Matriz de confusión SVM personas de perfil

	Feliz detectado	Triste detectado	Sorpresa detectado
Etiqueta feliz	57,4%	27,3%	15,3%
Etiqueta triste	21,81%	42,48%	35,7%
Etiqueta sorpresa	18,78%	38,12%	43,1%

Tabla 10. Matriz de confusión KNN personas de perfil

	Feliz detectado	Triste detectado	Sorpresa detectado
Etiqueta feliz	55,2%	29,12%	15,67%
Etiqueta triste	48,1%	31,4%	20,47%
Etiqueta sorpresa	52,35%	11,59%	40,85%

Tabla 11. Matriz de confusión RandomForest personas de perfil

	Feliz detectado	Triste detectado	Sorpresa detectado
Etiqueta feliz	61,46%	38,537%	0%
Etiqueta triste	45,3%	54,7%	0%
Etiqueta sorpresa	50,79%	49,21%	0%

5.3 Resultados delay sistemas embebidos

Acerca del apartado de los sistemas embebidos en este capítulo de resultados tenemos una comparación entre el *delay (*retardo) que se tenía en los sistemas embebidos, este retardo hace referencia a la imagen en la cámara, cuantos segundos después se veía reflejado en pantalla la acción que realiza el usuario.

En la figura 41 se pueden observar los resultados, donde la Raspberry tiene un retardo de 4 segundos y la NVIDIA Jetson Nano tiene un retardo de 2.5 segundos.

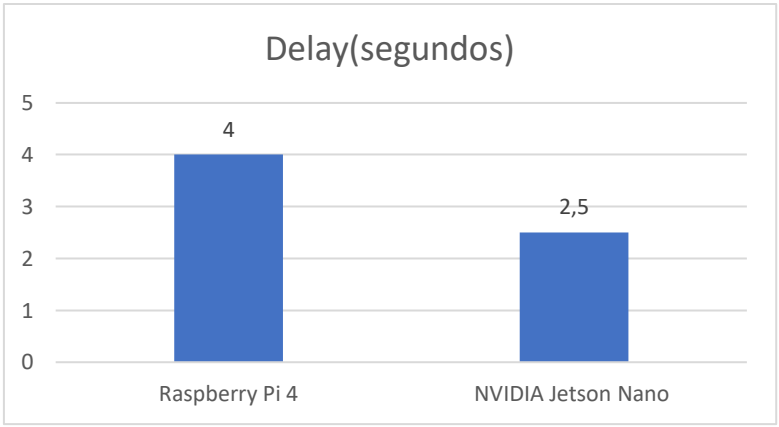

Figura 41. Delay en segundos sistemas embebidos.

5.4 Resultados almacenamiento de los datos

Como último apartado del capítulo de resultados se va a mostrar un ejemplo de cómo se guardan los datos, después de usar la aplicación, cuando se detiene se guarda un histórico de las emociones detectadas respecto a las personas que utilizaron el aplicativo, se genera un Excel con una tabla como la que se muestra en la figura 42.

	TimeStamp	Persona 1	Persona 2
0	2022-11-14 11:59:14	1	2
1	2022-11-14 11:59:14	0	0
2	2022-11-14 11:59:14	1	
3	2022-11-14 11:59:16	0	1
4	2022-11-14 11:59:17	1	2
5	2022-11-14 11:59:18	2	1
6	2022-11-14 11:59:18	1	0
7	2022-11-14 11:59:18	2	1
8	2022-11-14 11:59:18	1	2
9	2022-11-14 11:59:18	2	
10	2022-11-14 11:59:19	1	1
11	2022-11-14 11:59:19	0	
12	2022-11-14 11:59:20	1	
13	2022-11-14 11:59:21	0	
14	2022-11-14 11:59:22	1	
15	2022-11-14 11:59:22	2	
16	2022-11-14 11:59:22	1	
17	2022-11-14 11:59:22	2	
18	2022-11-14 11:59:23	1	
19	2022-11-14 11:59:23	2	
20	2022-11-14 11:59:23	1	
21	2022-11-14 11:59:23	0	
22	2022-11-14 11:59:24	1	
23	2022-11-14 11:59:25	2	
24	2022-11-14 11:59:25	1	
25	2022-11-14 11:59:25	2	
26	2022-11-14 11:59:25	1	

Figura 42. Recopilación de datos.

Capítulo 6

6.1 Análisis de resultados

Tras realizar todas las evaluaciones de las métricas para diferentes casos se procede a realizar un análisis de los resultados. En este capítulo se explicará la decisión sobre cual algoritmo de detección facial y de *Machine Learning* consideramos el más apropiado para la aplicación.

6.1.1 Algoritmo de detección facial

En cuanto al algoritmo de detección facial se tenían dos opciones, la primera es el modelo ya entrenado de *DLIB* el cual se explicó previamente, y la segunda es el algoritmo de detección facial de *HaarCascade*, en este apartado se tuvo en cuenta que algoritmo era mejor para implementar en un sistema embebido, como primera respuesta se tiene que el *HaarCascade* debido a su fácil implementación y su baja demanda de procesamiento pero el principal problema con este algoritmo es que no retorna una información concreta con lo que se pueda realizar un algoritmo de *Machine Learning* que no demande mucho procesamiento, para este caso no hay otra alternativa que usar una red neuronal.

Por otro lado el algoritmo de *DLIB* al retornar unos puntos (x,y) cabía la posibilidad de realizar algún cálculo matemático entre estos puntos y de esta manera realizar un algoritmo de *Machine Learning* de bajo procesamiento por esta razón, finalmente se decidió usar el modelo pre entrenado de *DLIB.*

6.1.2 Algoritmo de *Machine Learning.*

En cuanto al algoritmo de *Machine Learning* se tenían tres opciones: *Support Vector Machine* (SVM), *RandomForest y K-Nearest-Neighbor*(KNN).

Según los resultados obtenidos el primero en ser descartado es el algoritmo de KNN debido al bajo rendimiento en las pruebas realizadas respecto a los otros dos algoritmos, además de las métricas al probarlo en tiempo real este algoritmo no tenía una precisión aceptable para ser seleccionado.

Respecto al algoritmo de *Random Forest* se obtuvieron unos excelentes resultados cuando se trataba de una decisión binaria (feliz o triste) teniendo como pruebas escenarios ideales y no tan ideales tuvo unos excelentes resultados y al probarlo en tiempo real se obtenía un rendimiento óptimo, el problema con este algoritmo se presenta cuando se le agrega una tercera emoción, los resultados que retorna no son los esperados, su precisión llega a caer poco más del 40%, su coeficiente de Matthew's paso de ser 0.85 en hombres y 0.78 en mujeres a 0.385 en hombres y 0.37 en mujeres, por último, el puntaje F1 disminuyó aproximadamente 0.5 en cada resultado.

Finalmente, respecto al algoritmo de *Support Vector Machine(SVM)* obtuvo los mejores resultados en el proceso de evaluación de métricas, sin importar si eran escenarios ideales o no tan ideales se obtuvieron resultados por encima del 70% respecto a la precisión, por encima de 0.52 respecto al coeficiente de Matthew's y por encima de 0.73 respecto al puntaje F1, algo importante de recalcar es que los resultados entre los escenarios ideales (proceso A) y los escenarios complejos (proceso B) tienen una diferencia considerable, pero estos escenarios del proceso B se propusieron con el fin de exigir el sistema al máximo y observar que tan bien rendía con unas condiciones no tan optimas y este algoritmo fue el único que mantuvo unos resultados estables respecto a los otros.

Cuando se le agregó una tercera emoción se nota una superioridad de este algoritmo respecto a *KNN y RandomForest* manteniendo unas métricas aceptables, además de obtener unos resultados empíricos en tiempo real sobresalientes.

En conclusión, si la aplicación final del sistema va a ser una decisión binaria, es recomendable usar *RandomForest* debido a sus excelentes resultados en ese apartado además de su sencilla y

rápida implementación, por otra parte, si deja de ser binaria y se agrega una tercera opción de salida el mejor algoritmo en términos de rendimiento es *Support Vector Machine* aunque un contra de este es el tiempo del entrenamiento, dependiendo de la cantidad de parámetros que se varíen este tiempo aumenta exponencialmente.

6.1.3 Matrices de confusión

Respecto a las matrices de confusión obtenidas, primero se debe especificar a que corresponde cada fila y cada columna (tabla 12).

Tabla 12. Explicación campos matriz de confusión

Feliz verdadero	Falso feliz (detectó triste)	Falso feliz (detectó sorpresa)
Falso triste (detectó feliz)	Triste verdadero	Falso triste (detectó sorpresa)
Falso sorpresa (detectó feliz)	Falso sorpresa (detecto triste)	Sorpresa verdadero

Con esto claro, se va a analizar cada matriz por separado.

Respecto a la matriz obtenida del algoritmo SVM se puede observar que tiene unos excelentes resultados en la diagonal que son las predicciones acertadas, las predicciones falsas son menores a un 30% del total de datos analizados por lo cual esto da un gran nivel de confiabilidad para las predicciones realizadas por este sistema

Respecto a la matriz obtenida en KNN se puede observar que no tiene unos buenos resultados como se pudo apreciar de igual manera en las métricas del sistema obtenidas.

Finalmente, respecto a la matriz de *Random Forest* reafirma lo que se planteó en el análisis de los resultados de este algoritmo debido a que al agregar esta tercera posible salida del sistema no responde de manera correcta.

Respecto a las matrices de confusión vemos unos resultados mucho menos acertados para todos los algoritmos, pero de igual manera, el que se mantiene con los mejores resultados es el algoritmo de SVM, estos resultados se deben a lo complicado que es reconocer la expresión facial cuando una persona se encuentra de perfil debido a la poca información que le puede dar al sistema.

6.1.4 Sistemas embebidos

En este apartado se analizará cuáles son las ventajas y desventajas de cada sistema embebido.

Respecto a la Raspberry *PI* si bien observamos en la figura 41 tiene un retraso en la imagen mayor respecto a la *NVIDIA Jetson Nano*, esto se compensa un poco con la facilidad que se tuvo para la instalación de las librerías, este sistema operativo no dio muchos problemas para la instalación de las dependencias necesarias y la mayoría de sus instalaciones no superaban los dos minutos, este sistema embebido es ideal para proyectos que requieran una implementación rápida sacrificando un poco el rendimiento.

Por el lado de la NVIDIA *Jetson Nano* si bien se obtiene un rendimiento mucho mejor, la instalación de las dependencias tuvo varias complicaciones, dichas complicaciones se debían principalmente al tiempo de instalación, como primer ejemplo se tiene la instalación de *open-cv* que se tardó aproximadamente dos horas, este tiempo se debe a que dentro de la instalación para este sistema embebido se realiza la compilación de la librería para la GPU esto con el objetivo de obtener un mejor rendimiento, este sistema embebido es ideal para proyectos que requieran una capacidad de procesamiento alta y cuenten con el tiempo de realizar la instalación de todas sus dependencias, proceso que puede tardar más de un día de corrido.

Capítulo 7

7.1 Conclusiones

- La decisión sobre qué sistema embebido se va a utilizar depende de la aplicación y del usuario que lo va a manejar, si se necesita una implementación rápida la cual no genere impedimentos por temas de compatibilidad entre dependencias y sistema operativo lo adecuado sería usar la Raspberry Pi, otra ventaja de este dispositivo es que a usuarios que no tengan amplio conocimiento en Linux se les facilitan los procesos e implementaciones debido a que, como se mencionó anteriormente, no da problemas complejos en la instalación de dependencias, en cambio sí para la aplicación se busca el mayor rendimiento se debe trabajar con la NVIDIA *Jetson Nano*, con este dispositivo se necesita un mayor conocimiento sobre Linux debido a los diversos impedimentos que se encontrarán en el camino de la implementación, ya sea por instalación de dependencias, del sistema operativo o incompatibilidades.
- Una forma de optimizar los algoritmos y procesamiento de imágenes para que el programa se pueda adaptar a las condiciones de *hardware* limitado es bajar manualmente la resolución de las imágenes antes de ser procesadas y cuando se finalicé regresarlas a su resolución original; esto genera que el procesamiento sea más sencillo debido a que se reduce la cantidad de píxeles en los cuales los algoritmos de *Haar cascade* o *DLIB* tienen que identificar la presencia de un rostro en la imagen.
- El algoritmo pre entrenado de *DLIB* se le pueden hallar diversos usos dependiendo la aplicación, es un algoritmo el cual tiene una fácil implementación y un consumo de recursos moderado, al retornar las coordenadas del rostro reconocido se pueden realizar diferentes cálculos los cuales ayudarán a diseñar una analítica sin tener que hacer uso de *Software* avanzado.
- La mayoría de las bases de datos encontradas en el proceso de investigación están elaboradas de tal manera que las condiciones de ambiente sean perfectas como buena luz, buen enfoque, rostro de frente a la cámara y profundidad similar en todas las imágenes. Se evidenció que, para un mejor funcionamiento del sistema en tiempo real, con la cámara y rostro en movimiento es conveniente entrenar con una base de datos que se adecué a la realidad (imágenes a contraluz, rotación del ángulo de la cámara y rostros, desenfoque y objetos que obstruyan de la vista del rostro).
- De los tres algoritmos de *Machine Learning* probados dos de ellos obtuvieron buenos resultados, como primer decisión se descarta por completo el algoritmo KNN, esto debido a los deficientes resultados que obtuvo en las métricas, respecto a SVM y *Random Forest*, la decisión sobre cual usar viene a depender de la aplicación, si se va a trabajar en el diseño de un sistema que tenga que tomar una decisión binaria, que en el caso de este proyecto eran dos emociones, trabajar con *Random Forest* es la opción adecuada, aunque se debe tener en cuenta el factor aleatorio en el diseño de este sistema, cuando se tenga un buen resultado se debe guardar el predictor para no perderlo. Si la aplicación se escala a una decisión no binaria, la mejor opción por la cual optar es el algoritmo SVM, debido a su constancia en los resultados, rendimiento y fácil implementación.

7.2 Trabajo futuro:

- Realizar el historial de transacciones del cambio de estados en emociones en la nube con el fin de integrar el sistema a cualquier otra plataforma, con el fin de que la información no sea solo local o que se almacene en un archivo, por lo contrario, otros dispositivos puedan consultar el estado o cambios de emoción en las personas que se monitorean, para

así realizar un análisis de datos a través de un CMR como *Power BI* o exponer la información a través de un *API Gateway*.

- Implementar un sistema de toma de decisiones a partir de las emociones identificadas como: un control de acceso con identificación facial que permita el ingreso de las personas autorizadas que expresen con su rostro una emoción en específico.

Referencias

1. Liu, D., Bellotto, N. and Yue, S., 2020. Deep Spiking Neural Network for Video-Based Disguise Face Recognition Based on Dynamic Facial Movements. *IEEE Transactions on Neural Networks and Learning Systems*, 31(6), pp.1843-1855.
2. Jung, H., Lee, S. and Park, S., 2015. In: *Development of deep learning-based facial expression recognition system*.
3. Borges Oliveira, D., Ribeiro Pereira, L., Bresolin, T., Pontes Ferreira, R. and Reboucas Dorea, J., 2022. *A review of deep learning algorithms for computer vision systems in livestock*. [online] Science Direct. Available at: <https://www.sciencedirect.com/science/article/abs/pii/S1871141321003085> [Accessed 25 February 2022].
4. Mihai, D. and Valentin, A., 2020. In: *Automatic Traffic Sign Recognition Artificial Inteligence - Deep Learning Algorithm*.
5. *Real-time emotion recognition from facial images using Raspberry Pi II*. 2016.
6. CASTRILLON, WILLIAM A., & ALVAREZ, DAMIAN A., & LÓPEZ, ANDRÉS F. (2008). TÉCNICAS DE EXTRACCIÓN DE CARACTERÍSTICAS EN IMÁGENES PARA EL RECONOCIMIENTO DE EXPRESIONES FACIALES. Scientia Et Technica, XIV(38),7-12.[fecha de Consulta 3 de Marzo de 2022]. ISSN: 0122-1701. Disponible en: https://www.redalyc.org/articulo.oa?id=84903802
7. "Qué son los análisis de expresión facial y cómo funcionan", *Neuromarketing.la Información del sector para Latinoamérica*, 2022. [Online]. Available: https://neuromarketing.la/2016/12/los-analisis-expresion-facial-funcionan/#:~:text=1.-,Electromiograf%C3%ADa%20facial,capaz%20de%20brindar%20resultados%20precisos..
8. "Qué son los análisis de expresión facial y cómo funcionan", *Neuromarketing.la Información del sector para Latinoamérica*, 2022. [Online]. Available: https://neuromarketing.la/2016/12/los-analisis-expresion-facial-funcionan/#:~:text=1.-,Electromiograf%C3%ADa%20facial,capaz%20de%20brindar%20resultados%20precisos.
9. *Pang, Nianqiang, Li Facial expression recognition based on Gabor feature and neural network 2018. [Online]. Available:* https://ieeexplore-ieee-org.ezproxy.javeriana.edu.co/stamp/stamp.jsp?tp=&arnumber=8965443.
10. ["CommonLit | 3 datos sobre el poder de una sonrisa", *CommonLit*, 2022. [Online]. Available: https://www.commonlit.org/en/texts/3-datos-sobre-el-poder-de-una-sonrisa.
11. (2009). Por la expresión facial, ¿nos comunicamos?. *Revista Cubana de Enfermería*, *25*(1-2) Recuperado en 30 de octubre de 2022, de http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S0864-03192009000100001&lng=es&tlng=es.
12. Y. N. González-Meneses y J. Guerrero-García, «Análisis de bases de datos de expresiones faciales para la identificación automática de emociones centradas en el aprendizaje», *Rev. colomb. comput.*, vol. 22, n.º 2, pp. 58–71, dic. 2021.
13. FER2013 (21 de julio, 2020) FER2013 Recuperado de https://www.kaggle.com/datasets/msambare/fer2013
14. ckplus (21 de julio, 2020) ckplus Recuperado de https://www.kaggle.com/datasets/shawon10/ckplus
15. "K-Nearest Neighbors (KNN) in Python," JC Chouinard. [Online]. Available: https://www.jcchouinard.com/k-nearest-neighbors/.

Anexos

Repositorio de github https://github.com/JuanPZ2000/TG_ReconocimientoExpresionesFaciales