UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE ELECTRÓNICA
VALPARAÍSO - CHILE

DETECCIÓN DE EMOCIONALIDAD DEL PACIENTE EN CONSULTAS DE TELEMEDICINA ONCOLÓGICA

CRISTIAN ALBERTO BRUNA REYES

MEMORIA DE TITULACIÓN PARA OPTAR AL TÍTULO DE INGENIERO CIVIL EN TELEMÁTICA

PROFESOR GUÍA: MARCOS ZUÑIGA

PROFESOR CORREFERENTE: NICOLÁS TORRES

Agradecimientos

A mis padres, por apoyarme y motivarme a seguir en todo ámbito de la vida.

A mi hermano, por su amor incondicional.

A mis amigos de la universidad por su ayuda, apoyo y buenos momentos dentro y fuera de la USM.

A mis amigos de fuera de la U, por los carretes y noches jugando.

Y a mis abuelas.

Índice general

Índice de figuras

Índice de cuadros

1. INTRODUCCIÓN

En la actualidad existen diversas herramientas de software útiles para reuniones virtuales. Las consultas de telemedicina oncológica no se quedan exentas de su uso, facilitando esta actividad para las personas que no pueden asistir de manera física a la sesión. Como es de esperar, una consulta médica posee muchos factores que no se encuentran presentes dentro de la telemedicina, siendo uno de estos la interpretación de las emociones del paciente. Con este fin, esta memoria se centra en la utilización de algoritmos de machine learning para detectar seis emociones principales en el paciente; alegría, enfado, miedo, tristeza, sorpresa e indiferencia, además de determinar el nivel de atención del paciente. Considerando lo anterior, la detección de ritmo cardíaco es otro factor para potenciar la interpretación emocional. De esta forma, se busca mejorar la relación paciente-médico, generando un entendimiento emocional más eficiente hacia el paciente y así guiarlo en su tratamiento.

1.1. Contexto

Los profesionales de la salud, médicos oncológicos en particular, no tienen un contacto presencial con el paciente en las consultas telemáticas, por lo que, interpretar sus emociones se hace una tarea complicada al estar limitada al uso de una cámara y micrófono. Para solventar y llevar de mejor manera la teleconsulta es que la Fundación Arturo López Pérez (FALP) propuso el desafío de generar un sistema de detección de emocionalidad del paciente en tiempo real, entregando al médico las métricas asociadas y complementar así su interpretación de las emociones.

1.2. Definición del problema

Como sabemos que el paciente cuenta con una cámara de video en la sesión de telemedicina, la solución utiliza técnicas de visión por computador para que el paciente no deba realizar acción alguna y todo el procesamiento sea en paralelo a la teleconsulta. El software completo consta de un sistema de detección de ritmo cardíaco, detección de nivel de atención del paciente y el módulo de detección de emociones, además de la integración de estas tres características en una interfaz usuaria simple.

El foco de esta memoria se ecuentra específicamente en las últimas dos menciones: el módulo de detección de emocionalidad en tiempo real y la integración de los módulos en la interfaz usuaria.

1.3. Hipótesis

Es posible detectar en tiempo real, utilizando técnicas de visión artificial, al menos las seis emociones principales desde una sesión de telemedicina con una precisión superior al 75 %

1.4. Objetivos

1.4.1. Objetivo General

Desarrollar una herramienta de apoyo para la relación paciente-médico en telemedicina, y que así se logre una conexión emocional entre la reacción del paciente y el profesional de salud. Esto se realizará mediante la detección de emociones a través de la cámara del dispositivo de conexión del paciente.

1.4.2. Objetivos Específicos

1 Evaluar las mejores técnicas de detección de emociones según el estado del arte.

2 Desarrollar un sistema que detecte las emociones.

3 Detectar las seis emociones principales en el paciente: tristeza, ira, felicidad, miedo, angustia e indiferencia a través de la cámara.

4 Integrar los sistemas de reconocimiento de emociones, detección de ritmo cariaco y detección de atención en una API, de modo que se pueda acceder a ella durante la consulta.

1.5. Estructura de la Memoria

En el siguiente capítulo de este escrito veremos el *Estado del Arte*, donde se definen las técnicas, estrategias y aplicaciones del reconocimiento de emociones, y como estas son aplicadas en la industria actualmente. Además, tendremos la definición de los *datasets* investigados para realizar el entrenamiento adecuado en la utilización de técnicas de machine learning.

Luego, en el capítulo *Desarrollo de la Solución*, veremos el diseño de la solución que fue abordado junto a la FALP en primera instancia, y cómo este fue cambiando hasta la solución actual. Luego, en la subsección *Implementación* tendremos el desarrollo completo del módulo detección de emocionalidad y cómo fueron escogidos los métodos, el *dataset*, las variaciones del modelo y los resultados obtenidos. Además, tendremos el desarrollo e implementación de la *Interfaz Usuaria* como prueba de concepto, donde se incluyen los tres módulos de la solución completa.

Posteriormente se encuentra el capítulo *Validación*, donde están las pruebas realizadas al módulo detección de emociones. Finalmente se encuentra el capítulo de *Conclusiones* y trabajo futuro acerca del sistema y los módulos implementados.

2. ESTADO DEL ARTE

2.1. Modelos

Cuando dos personas entablan una conversación son capaces de entender la emocionalidad del otro más allá de lo dicho explícitamente, ver sus movimientos físicos, postura de cuerpo, tonalidad de voz, entre otros, pero esto se complica bastante en la interacción digital, específicamente en términos de interacción con cámara de vídeo. Como dice *Ali Alther, et al.* "One integral and necessary part of human behavior is emotion, which affects the way people communicate. Although human beings can recognize and interpret facial expressions, the identification of correct facial expressions continues to be a key and challenging task by computer systems"[2], que es justamente donde se enfoca esta memoria: ayudar a los sistemas computacionales a detectar emociones y hacerlo de la manera más eficiente posible.

En la actualidad existen diversas maneras de alcanzar este objetivo, siendo la más acertada utilizar modelos de aprendizaje de máquinas. Un modelo se presenta en el git denominado *FaceEmotion_ID* de *abhijeet3922* [1] utilizando redes neuronales profundas, donde plantea cuatro capas para clasificación de la cara detectada en píxeles de 48×48, donde predice una de las seis emociones planteadas según su capa de salida softmax. A este modelo se le conoce popularmente como *Mini_Xception* y su esquema se muestra en la figura 2.1.

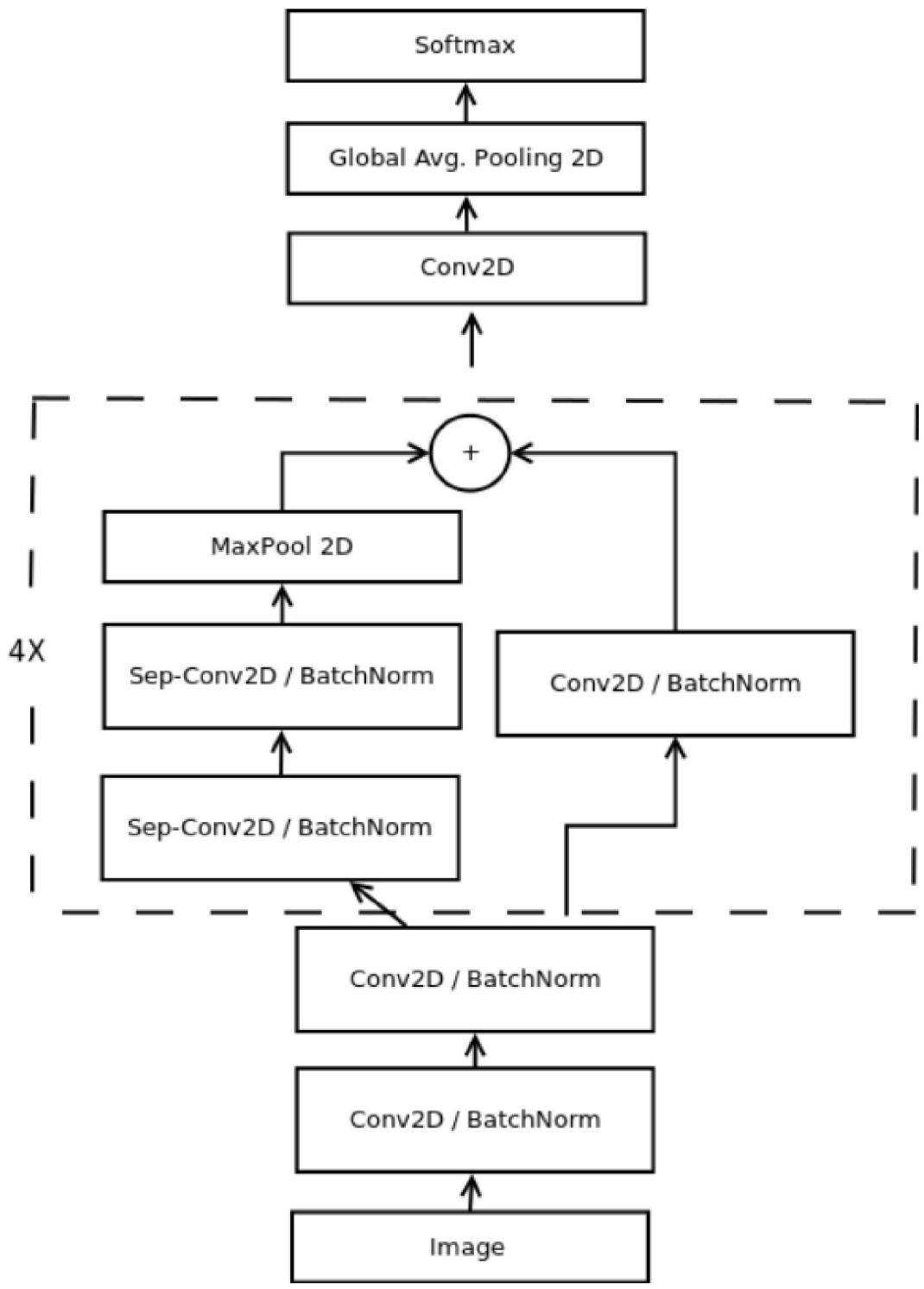

Fig. 2.1: Diagrama del modelo Mini_Xception

Por otro lado, tenemos a *KOUSTABH DAS* [6], quien en su kaggle nos presenta una variación leve del modelo anterior donde las capas iniciales son eliminadas y procesadas dentro del backbone. Esto genera un peor desempeño, específicamente tiene un $60\,\%$ de precisión utilizando el dataset *CK+48*, pero es una solución viable para problemas donde se requiera baja utilización de recursos de entrenamiento y/o procesamiento. El diagrama del modelo se muestra en la figura 2.2.

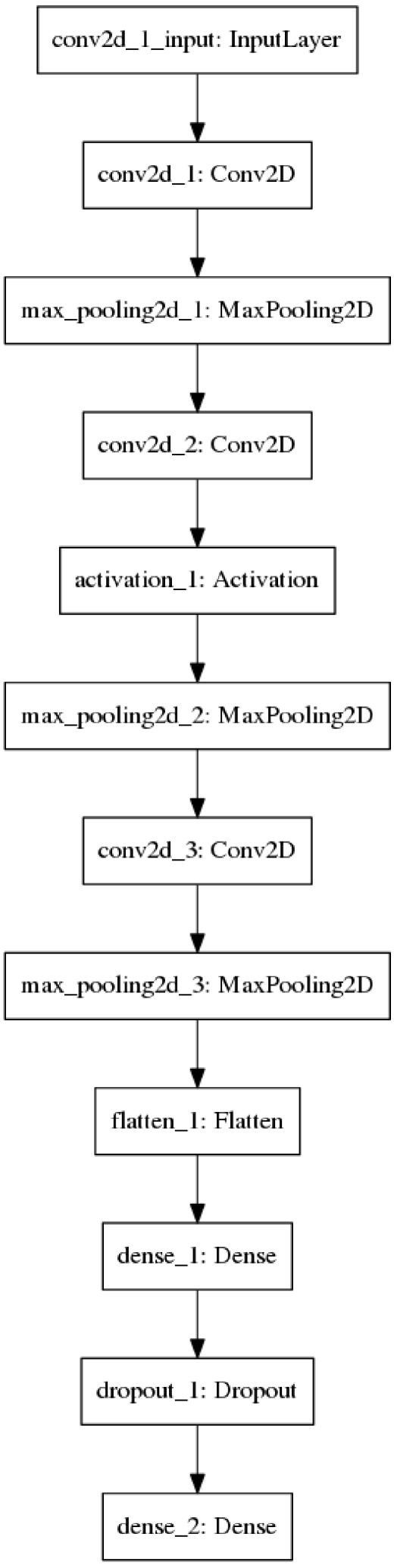

Fig. 2.2: Diagrama del modelo Human Emotion Detection

En el año 2019 se siguió investigando sobre la detección de emociones utilizando redes neuronales convolucionales (CNN), siendo uno de sus impulsores *Rohit Pathar, et al.*[7], quien en su investigación propone la estructura de la figura 2.3 donde se aprecia la cantidad de neuronas de la red planteada teniendo un total de ocho capas convolucionales, cuatro capas de max pooling y finalmente las tres capas de conexión completa. Este modelo presentó una precisión del $89\,\%$, pero tiene requerimientos elevados para su entrenamiento y utilización por su estructura.

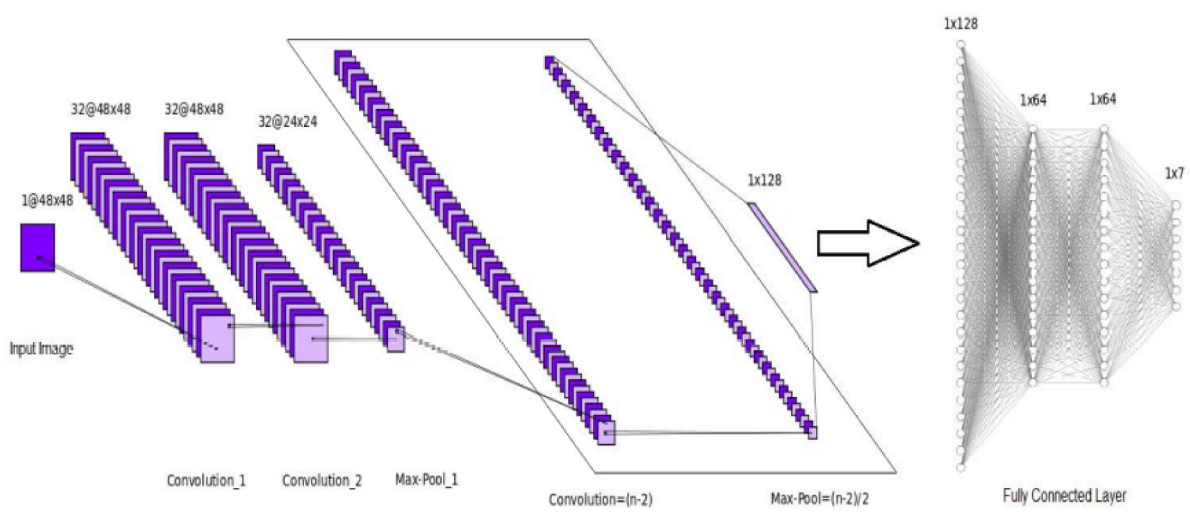

Figure 4: Architecture of deep network

Fig. 2.3: Diagrama del modelo reconocimiento de emociones humanas

Dentro de la detección de rostros de personas, se encuentran varios trabajos relacionados a la seguridad, como por ejemplo en el trabajo de Rajasekaran Thangaraj, et al. [9] se presenta la utilización de InceptionV3 para el procesamiento de imágenes provenientes de una cámara de CCTV donde se detecta las características faciales. La utilización de InceptionV3 para la detección de emociones, se evidencia en el trabajo de los autores del documento de AKM Nivrito [5] quienes realizaron un modelo utilizando InceptionV3. En este trabajo, se obtuvo una precisión significativa, el modelo era fácilmente entrenable y bastante acertado al ser probado en tiempo real. También *SANSKAR HASIJA*[8] realizó trabajos similares, pero utilizando una red *DenseNet169*.

2.2. Dataset

Cuando se menciona el tema de los conjuntos de datos utilizados para entrenar modelos de detección de emociones, uno de los más destacados y ampliamente reconocidos tanto en términos de cantidad como calidad de datos es el conjunto *Facial Expression Recognition 2013 (FER-2013)*. Este conjunto de datos, ilustrado en la figura 2.4, fue creado por *Goodfellow et al.*[10] y ha sido ampliamente utilizado en numerosas investigaciones para entrenar diversos tipos de modelos de aprendizaje automático relacionados con la detección de emociones, clasificación de géneros y características faciales en general. Su popularidad radica en su amplio alcance y calidad de los datos, lo que lo convierte en un recurso invaluable para la comunidad de investigación en este campo..

Fig. 2.4: Ejemplo datos FER-2013

Por otro lado, tenemos *KDEF and AKDEF (Karolinska Directed Emotional Faces and averaged KDEF)*, el cual consta de 4900 imágenes. Fue producido en el año 1998 por *Lundqvist, et al.*[11], pero es mayormente utilizado en la detección de intensidad de emociones,

atractividad y unas pocas emociones como son enojo o felicidad, esto debido a su estructuración, a la calidad de imágenes utilizadas (con un difuminado que a veces entorpece la detección) y a la etiquetación a través de una especie de codificación en el nombre de la imagen.

Fig. 2.5: Ejemplo datos KDEF

Finalmente, se investigó sobre el dataset *CK+* creado por *Patrick Lucey et al. in* [12] donde se etiquetan 327 vídeos faciales. Este conjunto de datos igualmente se utiliza en detección de emociones en modelos de prueba ligeros o llamados *"de juguete"* dado que los vídeos tienen cierto sesgo en la expresión representada por las personas.

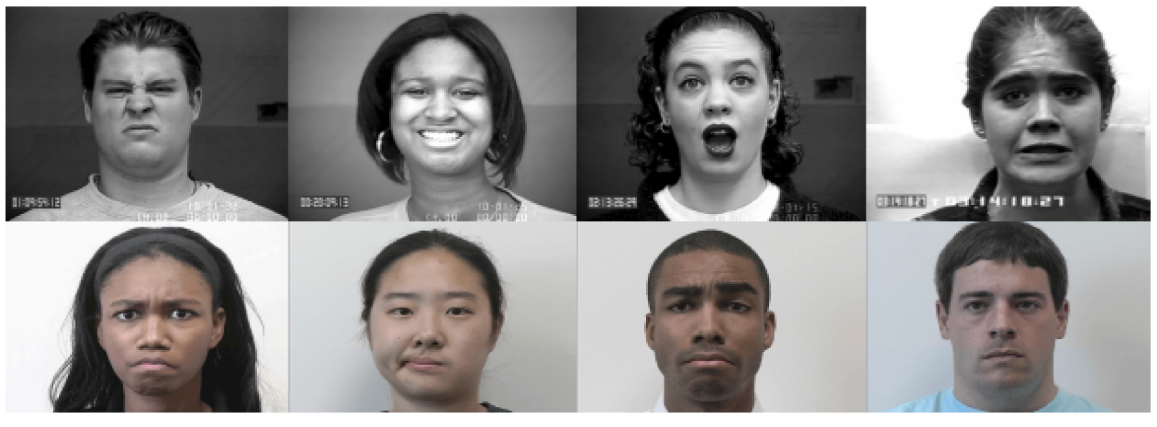

Fig. 2.6: Ejemplo datos CK+

2.3. Interfaz Usuaria

Cuando llega el momento de realizar la implementación de un modelo de machine learning para ser utilizado en un entorno real y no de pruebas, lo primordial es la facilidad de integración con el modelo. Es por esto, que el primer pensamiento es desarrollar utilizando el lenguaje de programación *Python*, dado que el modelo está creado utilizando Keras. En esta línea tenemos una gran cantidad de *frameworks* disponibles para el desarrollo, donde destaca *Django*[14] por su trayectoria y amplia comunidad creada desde el año 2003, pero pese a esto, su velocidad de desarrollo se ve limitada por la utilización de sus modelos reusables, lo cual también complica la actualización entre versiones del framework. Por otro lado, tenemos a *Flask*[15] el cual permite un desarrollo fácil y amigable, pero requiere y utiliza bastantes módulos de terceros, lo cual puede ser un problema de seguridad, además de ser bastante más lento dado que las consultas son secuenciales siempre, lo que requiere un procesamiento mayor. Finalmente, tenemos a *Fast Api*[16] el cual, como su nombre lo indica, es de los frameworks basados en Python más rápidos tanto para desarrollar como en la utilización. Su único contra es la baja comunidad al ser relativamente nuevo, pero esto se contrarresta con su buena documentación.

2.4. Discusión

Con todos estos elementos investigados y previamente mencionados, se llegó a que la mejor opción para aplicar al problema es utilizar el modelo InceptionV3 y agregar variaciones, para tratar de optimizarlo y subir el porcentaje de precisión. Por otro lado, el dataset a utilizar será FER-2013, dada su amplia utilización y documentación, además de su estructura y simplicidad al momento de utilizar. También se definió que el framework Python a utilizar será Fast Api, dada su velocidad y la futura implementación de esta memoria en teleconsultas en tiempo real, lo cual no entorpecería la velocidad y estabilidad de la llamada.

3. DESARROLLO DE LA SOLUCIÓN

3.1. Análisis

En una primera iteración, se trabajó semanalmente junto a la FALP en el diseño del sistema. Destacar que la contraparte no contaba con un sistema similar o idea preconcebida de cómo querían el sistema. El resultado del diseño para esta primera iteración se grafica en el diagrama de la figura 3.1

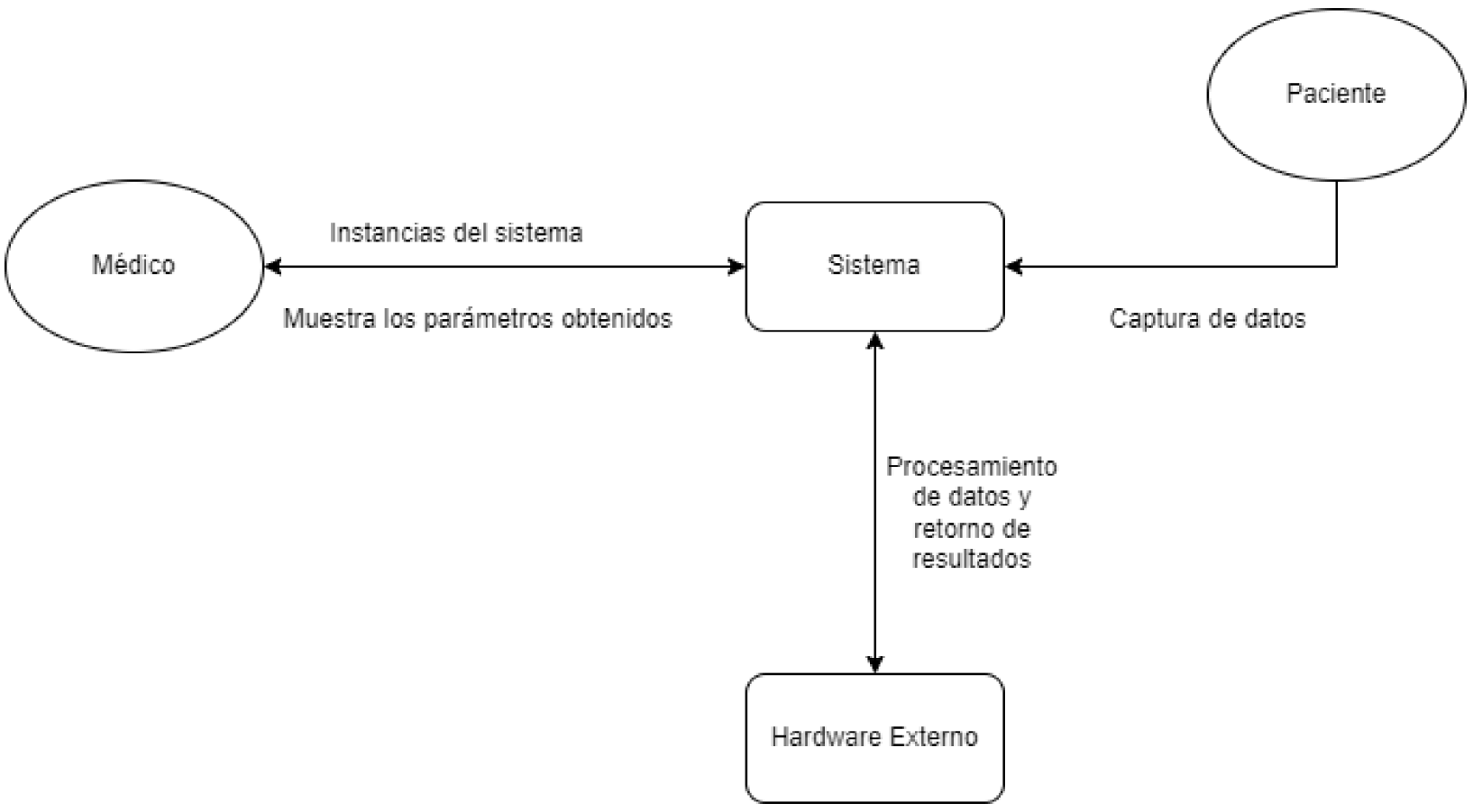

Fig. 3.1: Diagrama de contexto 1

Se aprecia que, al iniciar la teleconsulta, esta se realiza a través del sistema donde se recolectan los datos del paciente (vídeo cámara). Estos datos son procesados obteniendo así las métricas de emocionalidad, bpm y nivel de atención, para luego ser reenviadas únicamente

al médico para ayudar y complementar el entendimiento del estado del paciente.

Luego se dio paso a la segunda iteración del sistema, donde se plantea un procesamiento en paralelo, independiente a la teleconsulta, dado que así se puede modularizar el servidor de procesamiento:

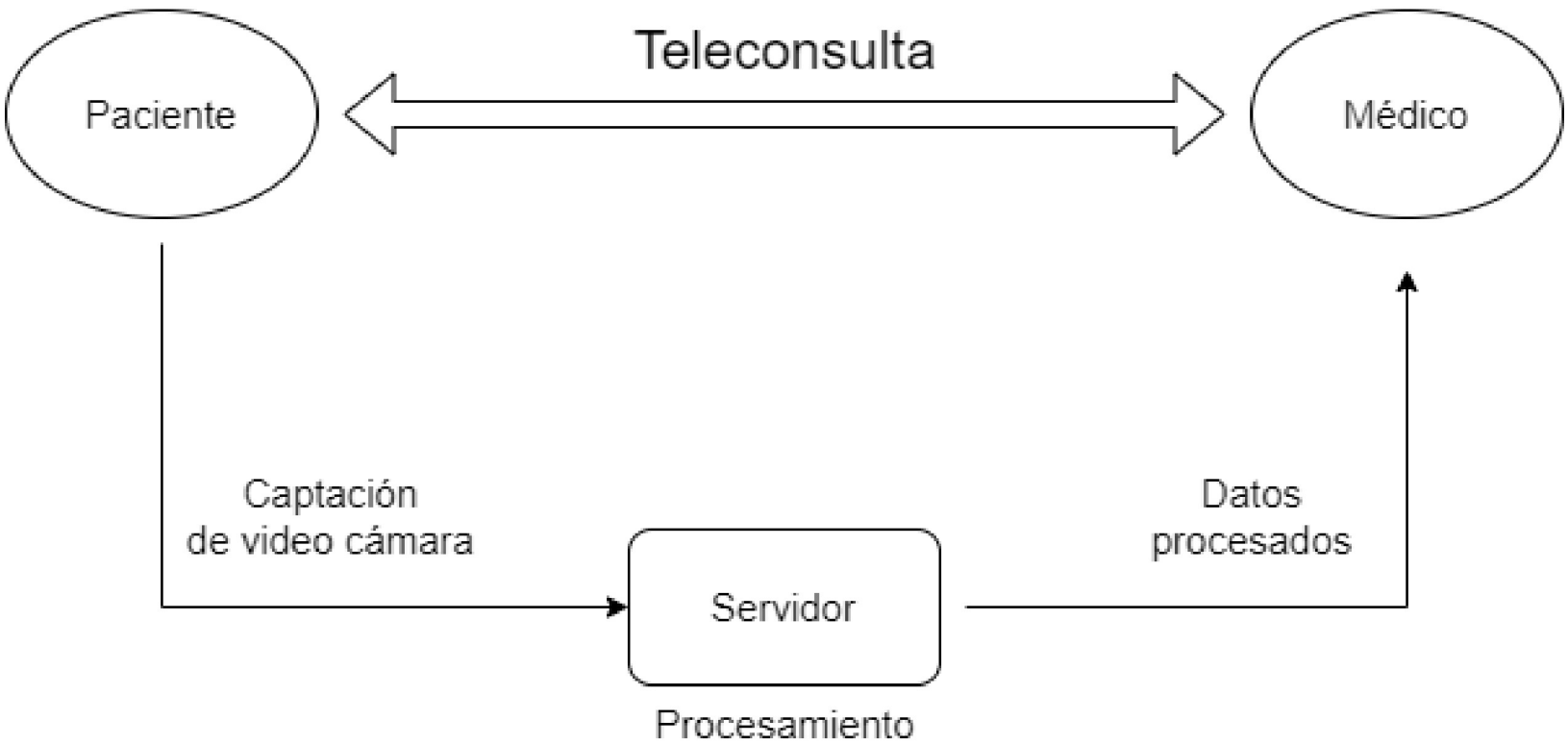

Fig. 3.2: Diagrama de contexto 2

El resultado de diseño en esta iteración fue el seleccionado por la contraparte para llevar a implementación, dado que especifica un procesamiento independiente de su sistema actual de teleconsulta, donde ellos podrían capturar directamente la pantalla y extraer el vídeo sin problemas para ser enviado a sus servidores de procesamiento, evitando entorpecer la llamada.

3.2. Diseño

Luego de la fase de análisis del problema y habiendo definido como se implementaría la solución, se dió paso a la fase de diseño junto a la FALP, donde se definió cómo se realizaría el desarrollo del sistema, dando como resultado el siguiente esquema:

Fig. 3.3: Diseño por módulos

En el marco del Programa de Memorias Multidisciplinarias, el esquema de la figura 3.3 especifica la modularidad del sistema en la fase de diseño: *Emocionalidad*, que se encarga de detectar la emoción del paciente en tiempo real y donde se centrará el contexto de esta memoria; *Ritmo Cardíaco*, que se encarga de detectar el bpm del paciente utilizando rPPG (Fotopletismografía remota); y finalmente *Detección de Atención* que se encarga de porcen-

tualizar el nivel de atención que el paciente está prestando durante la teleconsulta. Todos estos módulos quedarán incluidos en una interfaz de usuario que despliega la información para el médico en forma de texto simple.

3.3. Implementación

3.3.1. Detección de emociones

Datasets

De los dataset investigados hay dos principales, los cuales son: FER-2013 y KDEF/AKDEF. Se definirá cada uno y sus conclusiones en las secciones siguientes. Sin importar su cantidad de emociones clasificadas o su resolución de imágenes, la FALP solicitó enfocarse en seis emociones principales, las cuales son: Felicidad (Happy), Enojo (Angry), Sorpresa (Surprise), Miedo (Fear), Neutral (Neutral) y finalmente Tristeza (Sad).

- **KDEF/AKDEF.** Consta de 4900 imágenes de expresiones faciales humanas. El dataset incluye 70 personas distintas realizando siete emociones cada uno en resolución 562x762 píxeles. Su estructuración no es óptima para realizar las pruebas de manera sencilla sin preprocesar los datos, por lo que quedó descartado por temas de tiempo y la amplitud del proyecto, pero es posible utilizar y queda como trabajo futuro.

- **FER-2013.** El dataset incluye 28.709 imágenes en resolución 48x48 píxeles en training y 7.178 en testing lo que nos da una relación 80-20 óptima. Además, su estructuración y etiquetado vienen listos para ser utilizados. Consta de siete emociones, donde, además de las emociones solicitadas por la FALP, se encuentra la emoción de *Disgusto*. Sin embargo esta emoción dispone de una muy baja cantidad de datos, lo que la hace irrelevante para el análisis.

Imágenes de ejemplo del dataset:

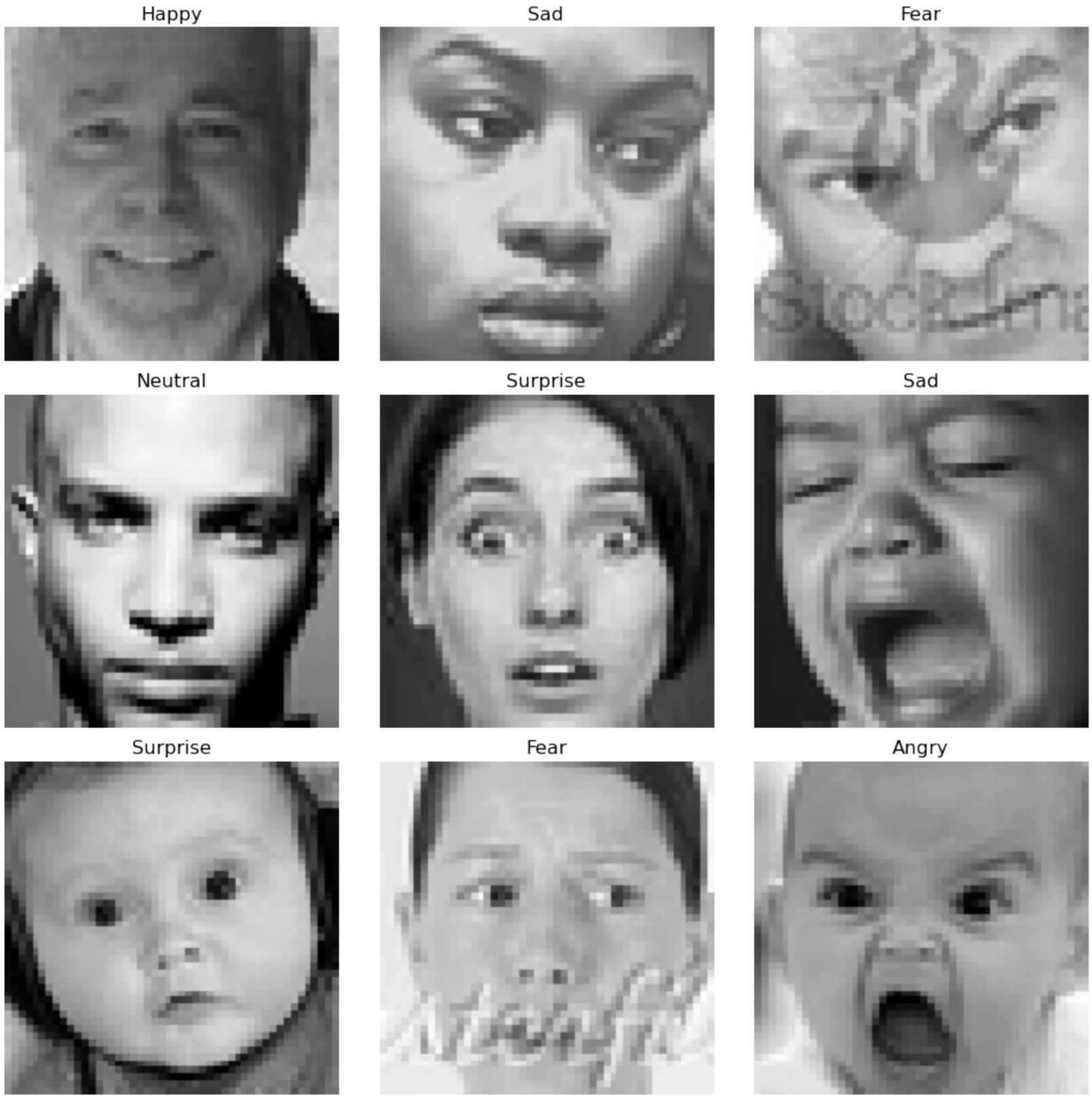

Fig. 3.4: Imágenes de ejemplo FER-2013

Destacar que tampoco es un dataset 100 % limpio, por que hay imágenes como la de arriba a la derecha (Fear) o abajo en el centro (Fear) que cuentan con textos o *ruido* dentro de la misma imagen, que pueden interferir en el entrenamiento del modelo.

Luego tenemos la distribución de datos en el sub-dataset de entrenamiento:

Fig. 3.5: Distribución en FER-2013

Se nota cierto equilibrio en el dataset, exceptuando por disgusto que cuenta con la menor cantidad de imágenes, pero tampoco es una emoción pedida por la contraparte dado que no es una emoción frecuente en la reacción de los pacientes a los diagnósticos dados en las teleconsultas en el ámbito del cáncer.

Se planteó un tiempo realizar *Data Augmentation*, dada la cantidad de datos, y tratando de mejorar el algoritmo 1, pero fue descartado luego de una semana por su entrenamiento, donde su rendimiento bajaba al rededor de un 20 %.

En conclusión, el dataset cumple con las expectativas para ser utilizado en el entrenamiento de los algoritmos a probar, sobre todo por su estructuración y simpleza, incluida la distribución 80-20 que este cumple.

La carga de imágenes del dataset en el algoritmo se realizó a través de la función *flow_from_directory* de *Keras* ya que, al venir los datos ordenados y etiquetados en directorios, esta función mantiene la jerarquía de los datos, haciendo así más sencillo su tratamiento.

Además, previamente se utilizó un *ImageDataGenerator* para realizar un leve *aumento de datos (data augmentation)* con transformaciones simples de rotaciones, zoom y reescalados, para así obtener un mayor volumen de datos y variaciones.

Destacar que para el segundo modelo, que se definirá en profundidad más adelante, se realiza un preprocesamiento que incluye Keras para *InceptionV3*, el cual se define como *inception_v3.preprocess_input* donde se reescalan los píxeles entrantes de las

imágenes al intervalo [-1, 1], para evitar el problema de *desvanecimiento del gradiente* en las funciones de activación de los modelos.

Algoritmos

Para realizar la detección de emociones se hizo una fase de validación de parámetros con dos algoritmos distintos. A continuación se detallará cada implementación y sus conclusiones.

- **Algoritmo 1: Detección de Emociones utilizando CNN.**

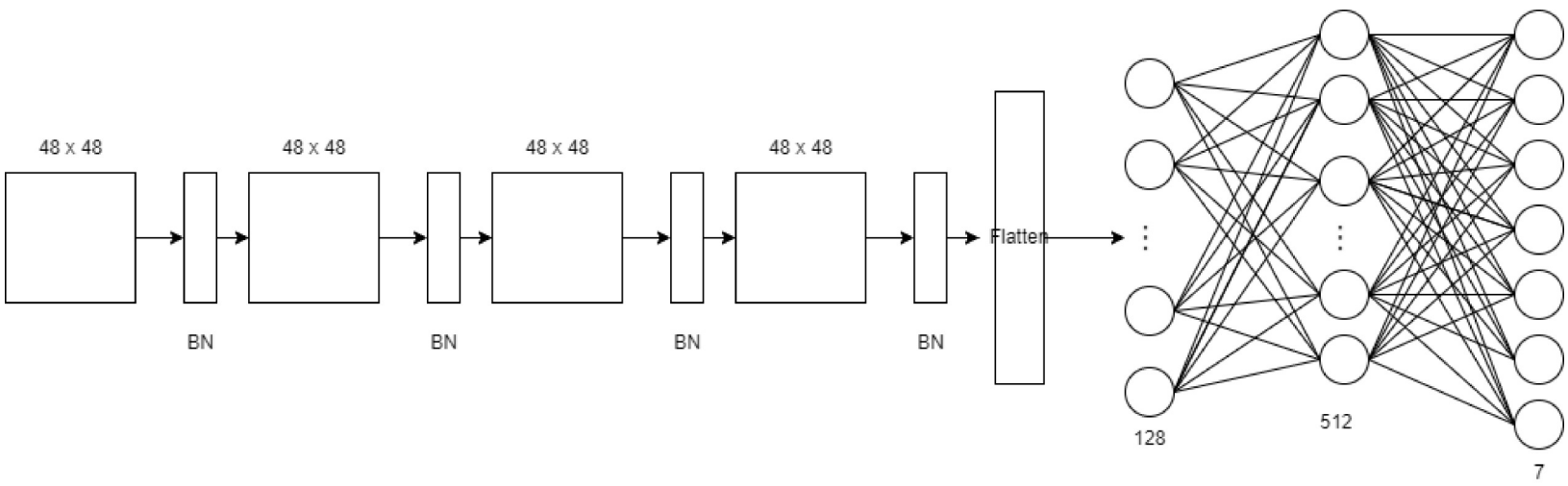

Fig. 3.6: Diagrama Modelo 1

Para la generación de este algoritmo, se generan cuatro capas convolucionales con activación *relu* con un *input shape* del tamaño de las imágenes del dataset (48 x 48) seguidas de un *BatchNormalization*. Luego se implementa una capa FNN (*feedforward neural network*) para la conexión con la capa de salida, la que consta de siete neuronas con función activación *softmax* (una por cada emoción).

Luego de hacer un entrenamiento con optimizador *adam* y 50 épocas, se obtienen los siguientes resultados:

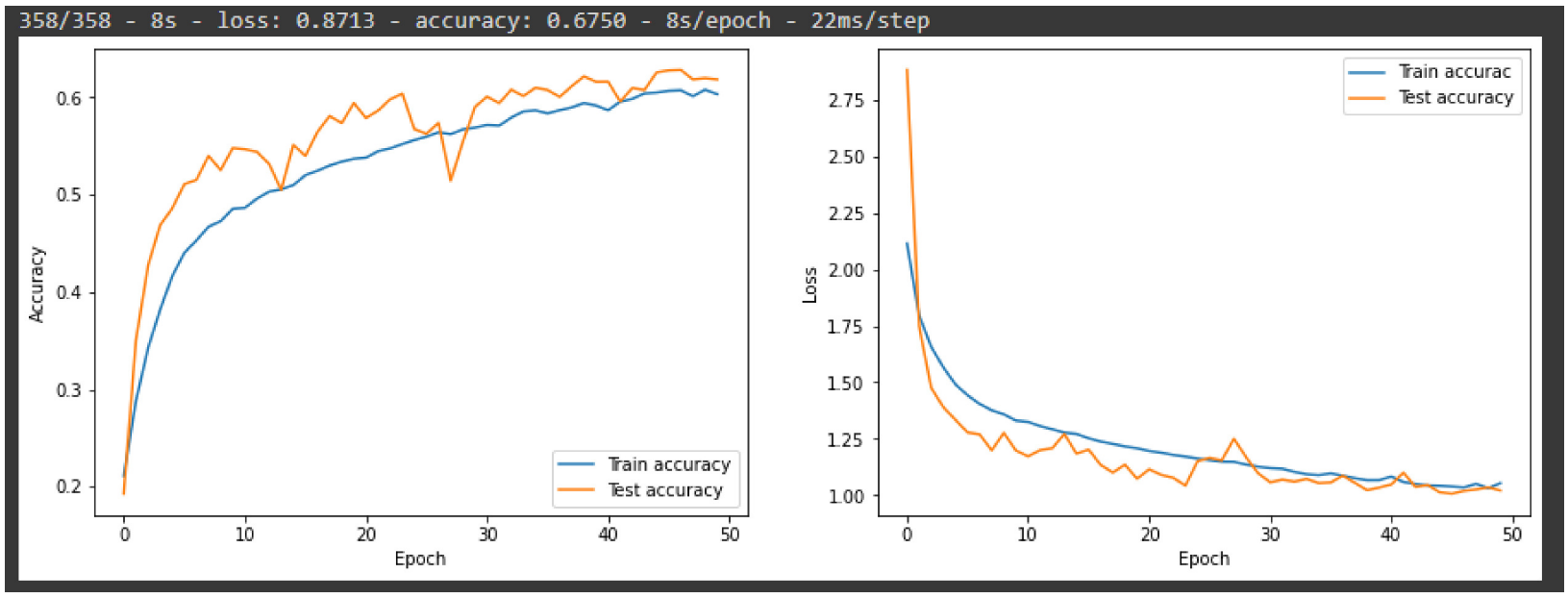

Fig. 3.7: Training Modelo 1

Notamos que obtuvo un training accuracy de 67 %, mientras que su loss es de 87 % por lo que procedemos a realizar las pruebas con el set de validación del dataset *FER-2013*, generando la matriz de confusión y el gráfico ROC-AUC siguientes:

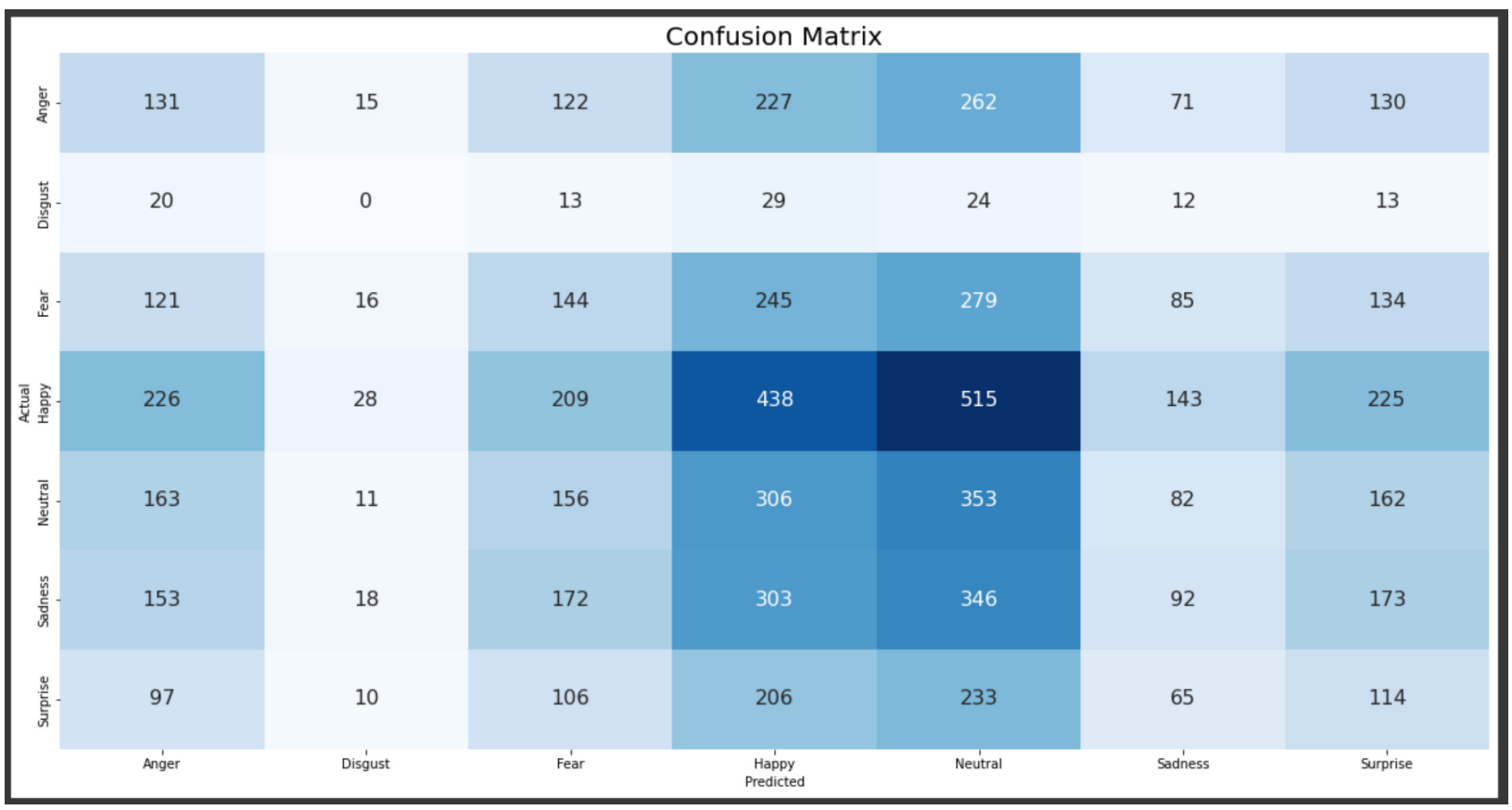

Fig. 3.8: Matriz de confusión Modelo 1

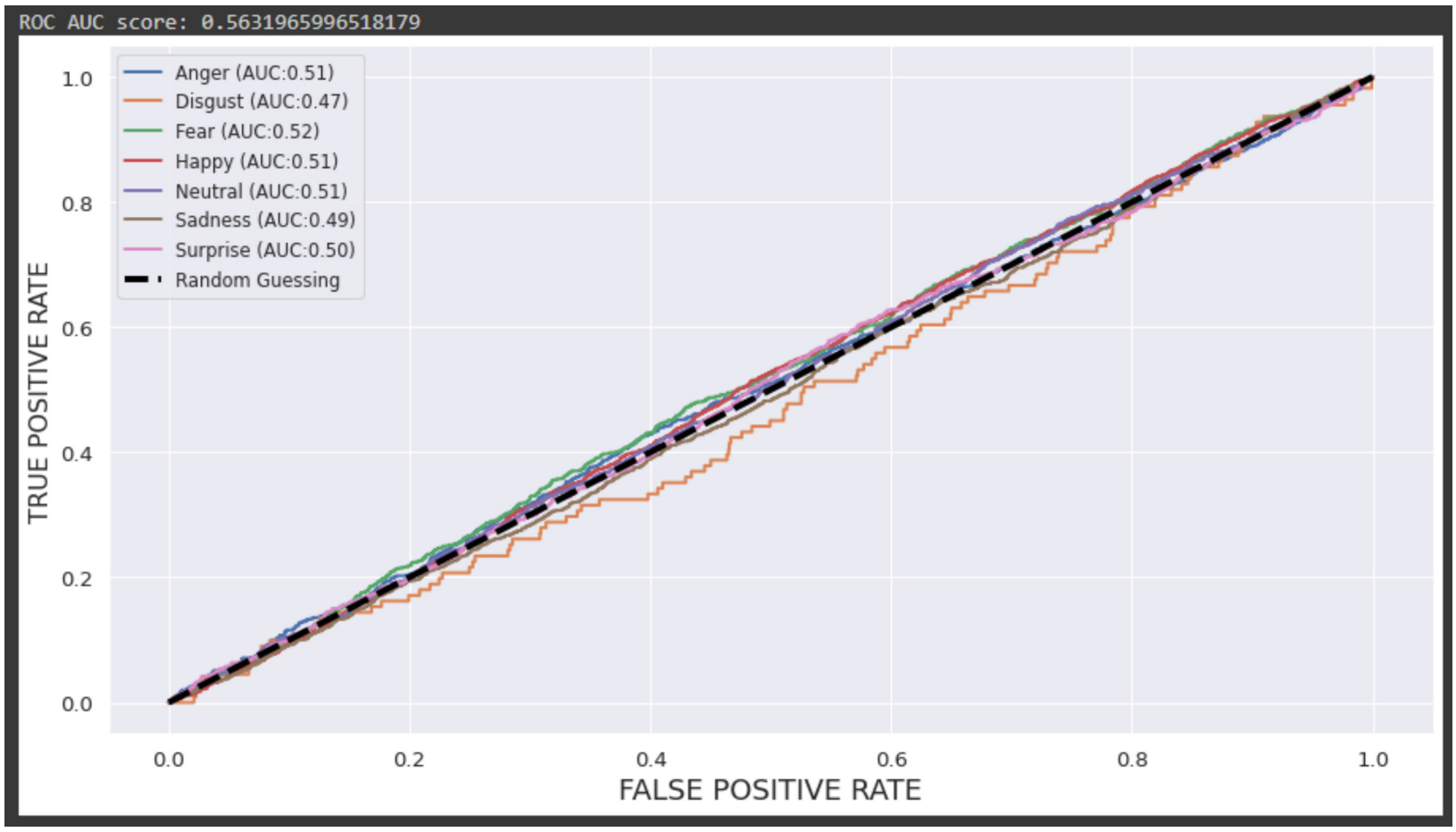

Fig. 3.9: ROC-AUC Modelo 1

Se observa en la matriz de confusión, resultados bastantes deficientes, por un número significativo de falsos positivos y falsos negativos. Destacar que de igual manera se nota, la falta de datos en la emoción *Disgusto*.

Se utiliza la curva *ROC* como parámetro secundario para evaluar la eficacia del algoritmo dado que, como los ejes lo indican, es la relación entre la tasa de falsos positivos (fpr) y verdaderos positivos (tpr). Destacar que *AUC* es el área bajo la curva y el puntaje de *AUC* obtenido indica qué tan bien funciona el algoritmo. Al ser multiclase, tenemos un AUC individual por cada emoción (tabla superior izquierda del gráfico) y el promedio de estos da el *ROC-AUC score* posicionado en la parte sobre el gráfico. Las curvas obtenidas son prácticamente una diagonal para todas las emociones, lo que indica que los resultados obtenidos son prácticamente lanzar una moneda.

- **Algoritmo 2: Detección de emociones utilizando InceptionV3.**

  Inception es una red neuronal convolucional utilizada en análisis de imágenes y detec-

ción de objetos que cuenta con una exactitud superior al 78,1 % en conjuntos de datos de ImageNet. Este modelo está compuesto por una secuencia de capas convolucionales, *MaxPool, Concat, Fully connected*, entre otras, proponiendo en esta arquitectura los bloques inception. La siguiente figura muestra un diagrama de alto nivel del modelo:

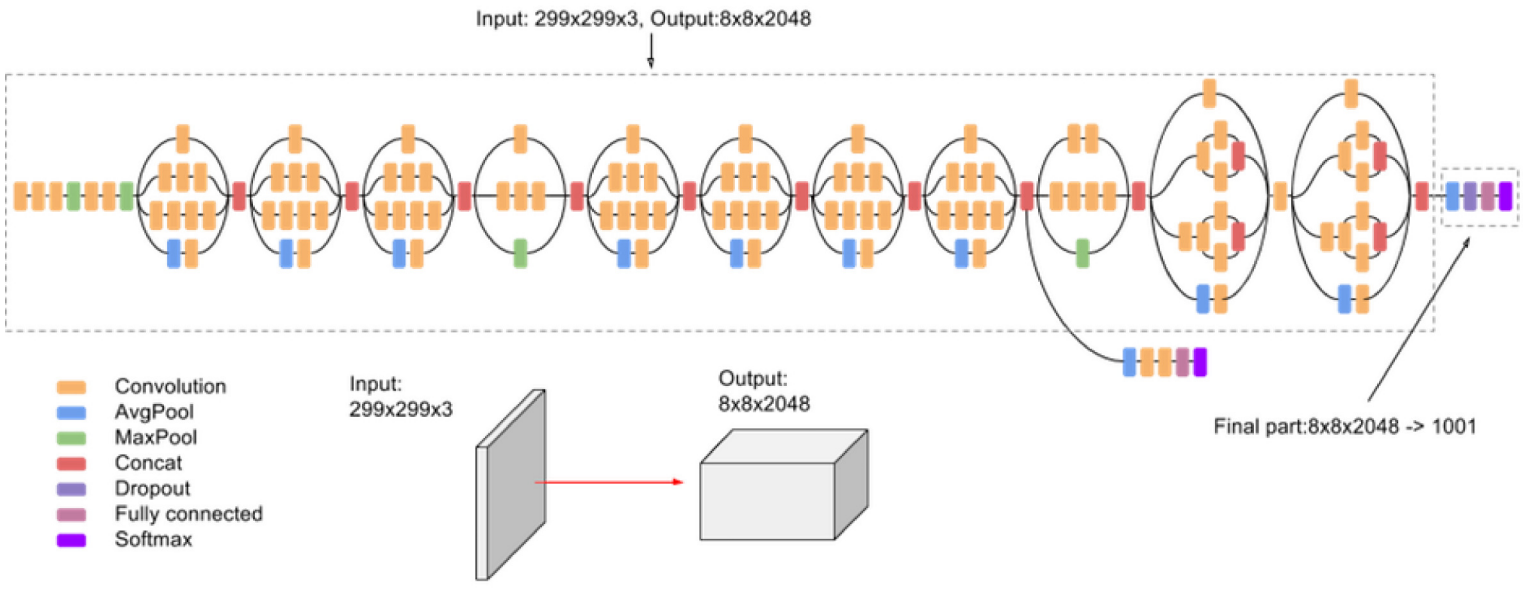

Fig. 3.10: Diagrama modelo 2

Como toda arquitectura de deep learning, para la adaptación de este problema a través de transfer learning, se utiliza la arquitectura InceptionV3 suprimiendo las capas fully connected (marcadas como *Final Part* en el diagrama) y se reemplazaron por una capa *Pooling2D*, una capa densa de 1024 neuronas y finalmente la capa de salida con siete neuronas en *Softmax* para que se ajuste a nuestro problema de detección de emocionalidad. Este modelo fue entrenado con 60 épocas, y un *Early Stopping* de *paciencia* 5, el cual se detuvo en la época 40. Luego se aplicó un *fine tuning* de 20 épocas.

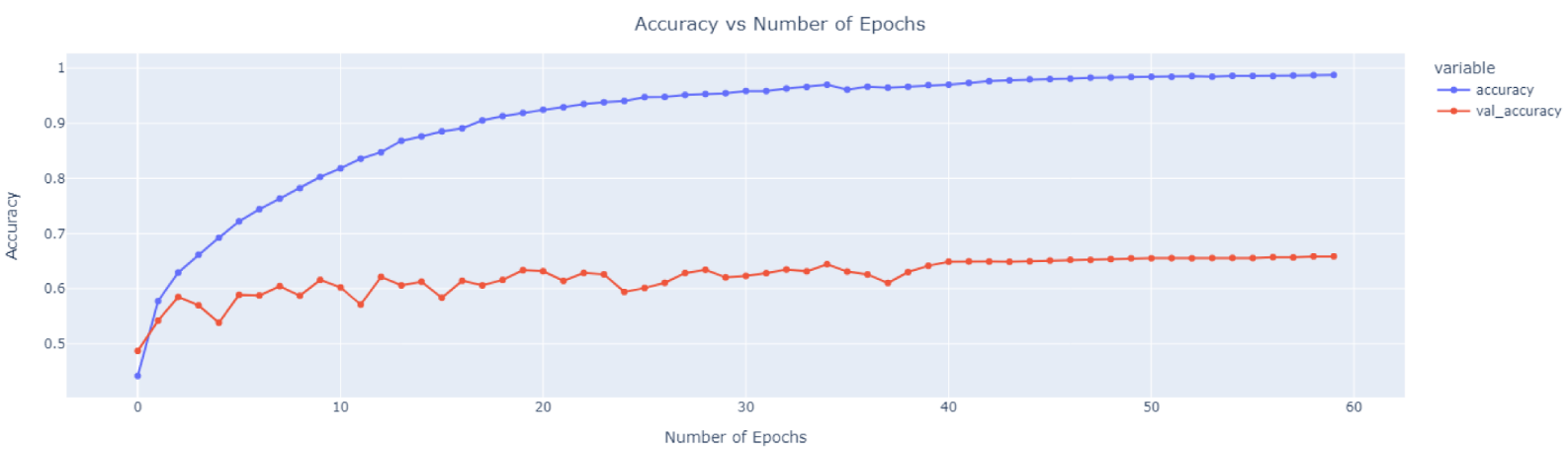

Fig. 3.11: Training modelo 2

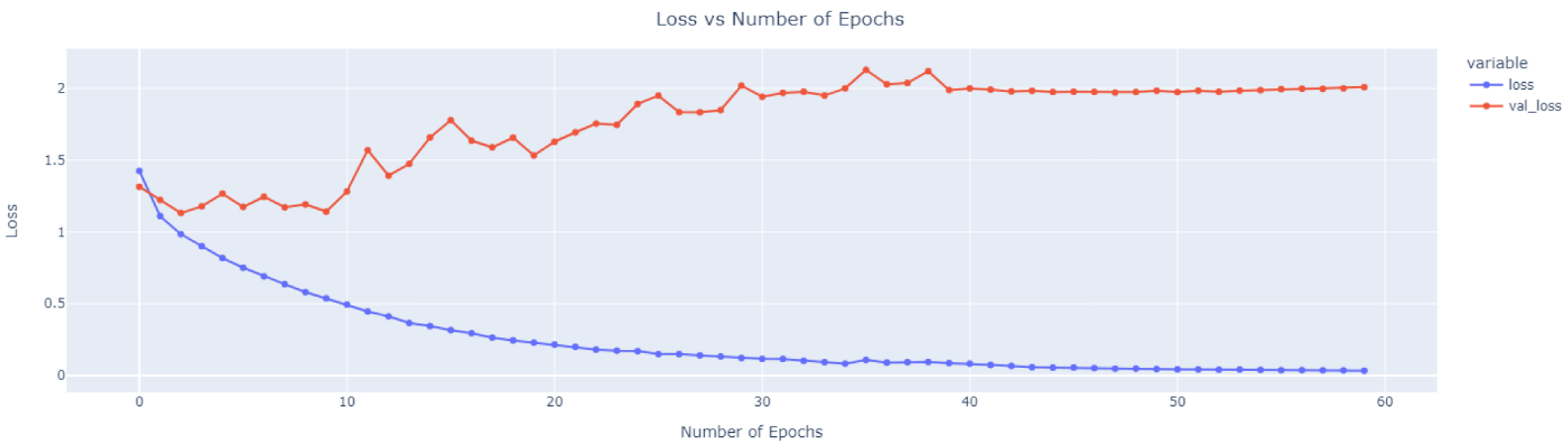

Fig. 3.12: Training modelo 2

El valor de training accuracy fue de un 98.7 %, lo cual es un incremento de un 31 % respecto al modelo anterior. Además, su curva de entrenamiento es mucho más suave, lo que indica una clara estabilidad al momento de entrenar. Pese a ello, el valor *val_accuracy* nos muestra un estancamiento luego de la época 15 aproximadamente lo que significa que el algoritmo dejó de aprender y ocurrió *Overfitting* durante el proceso de aprendizaje, lo cual podría ser perjudicial para la fase de predicción con instancias particulares en relación al set de entrenamiento.

Luego se obtienen las métricas de matriz de confusión y *ROC-AUC* siguientes:

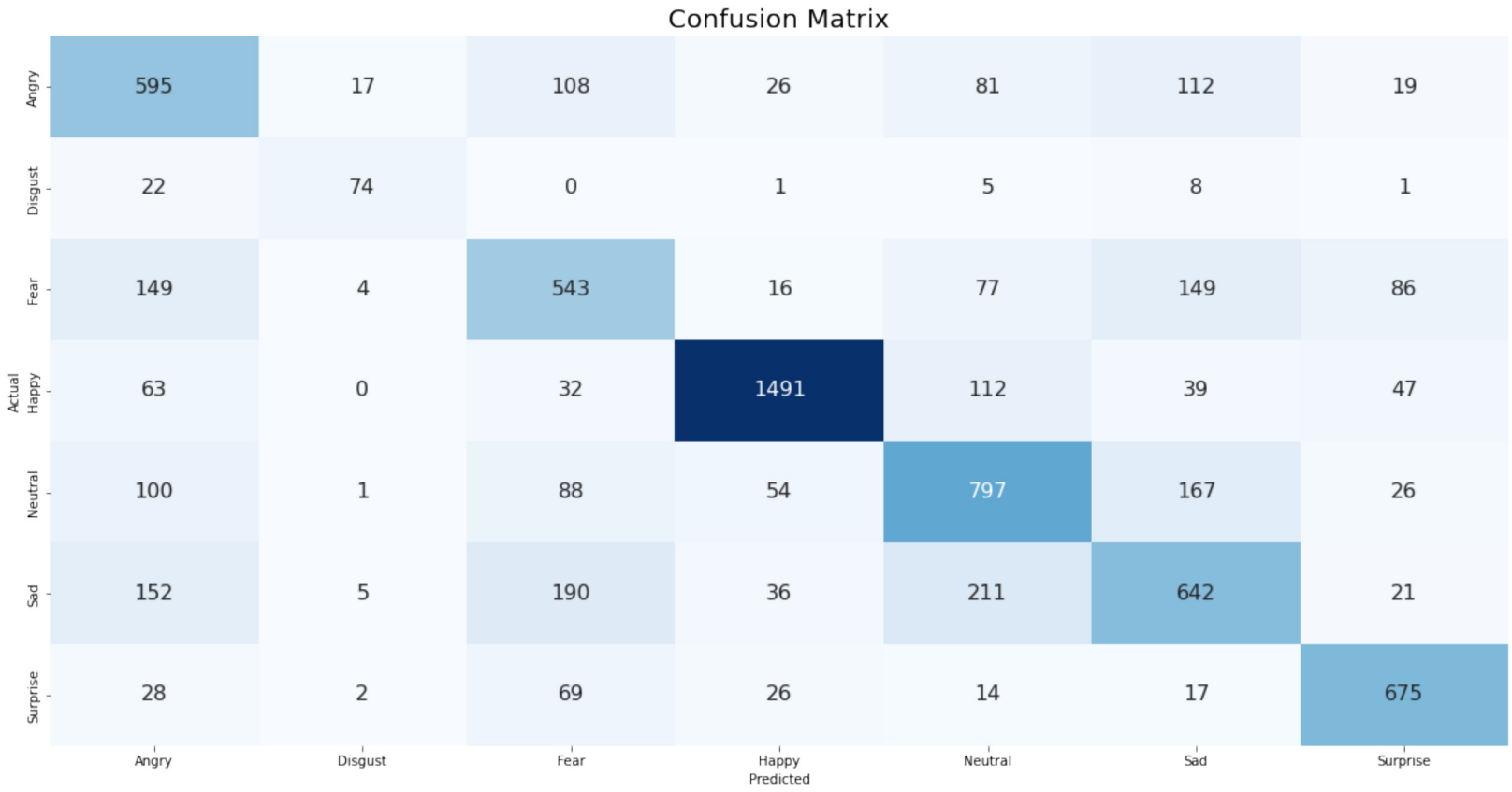

Fig. 3.13: Matriz de confusión modelo 2

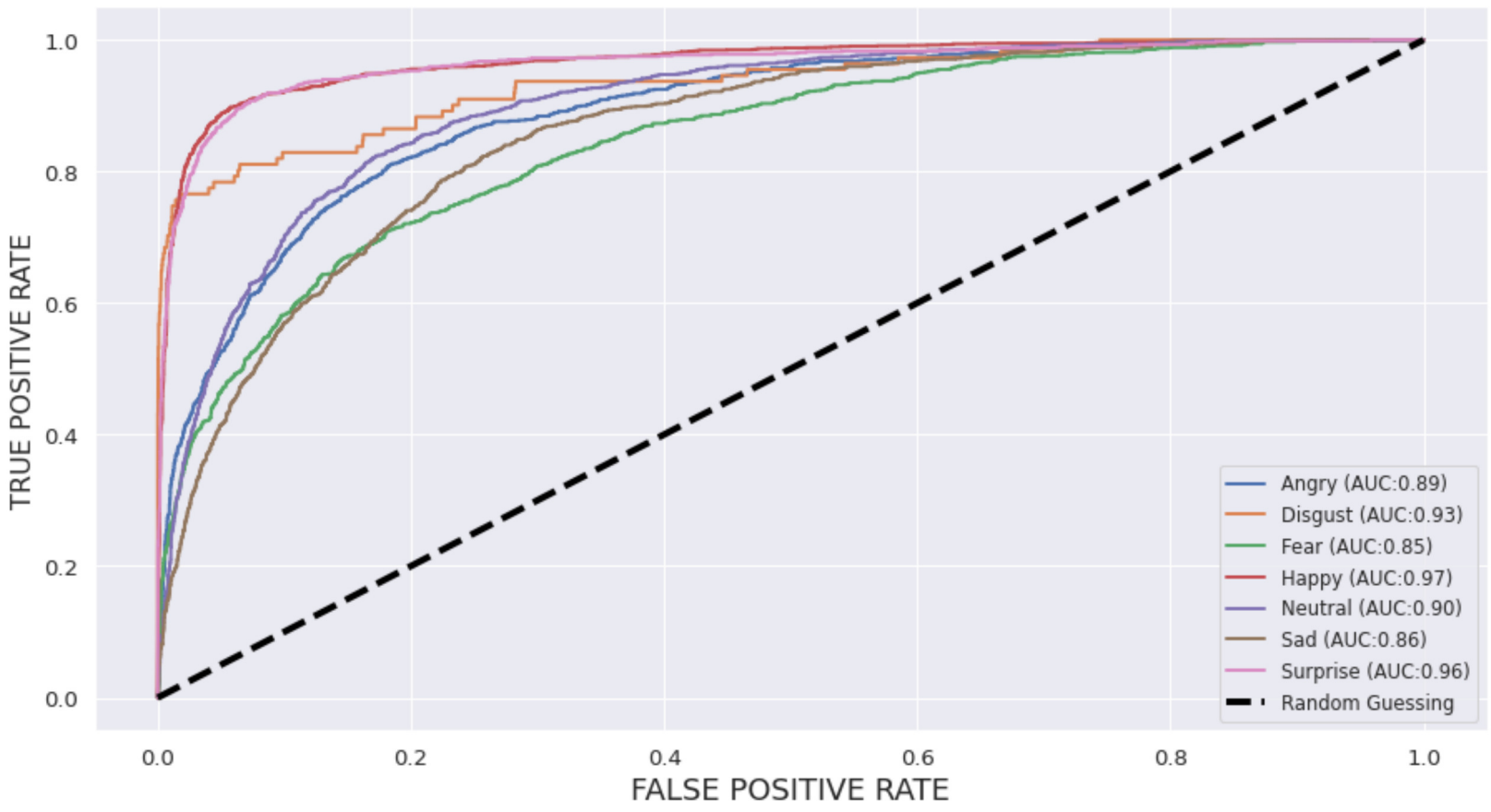

Fig. 3.14: ROC-AUC modelo 2

En este caso, la matriz de confusión muestra un comportamiento de este modelo superior al modelo presentado anteriormente, detectando de manera eficaz e independiente

las emociones y no hay confusión entre ellas al momento de detectar alguna. Por otro lado, notamos que el valor de *ROC-AUC* es 0.922, es decir, detecta con precisión un 92.2 % de las emociones, siendo la peor detectada el Miedo, mientras que la con mejor reconocimiento es Felicidad con un 97 %.

Variaciones

Se tomaron en consideración variaciones del algoritmo, ajustando ciertos parámetros y características de la red que se detallarán a continuación:

- **Variaciones de algoritmo 1.**

  Las variaciones del algoritmo 1 fueron principalmente en las capas del algoritmo, es decir, agregando Dropouts y Maxpool2D, dado que el resto de variaciones no eran tan notorias en general para el valor de accuracy ni en las metricas de ROC-AUC.

Tab. 3.1: Variaciones realizadas al algoritmo 1

Variación	Capas	Optimizador	Loss	Accuracy
1	4 capas + salida	Adam	catregorical_crossentropy	0,6750
2	4 capas + salida + Dropouts + Maxpool2D	Adam	categorical_crossentropy	0,699
3	4 capas + salida + Dropouts	Adam	categorical_crossentropy	0,7377

- Para los nombres y links de archivos ver anexo 1.

Pese a estas variaciones, el algoritmo seguía respondiendo de la misma manera, es decir, con matrices de confusión deplorables y un ROC-AUC alrededor de 0.55 para todas las variaciones.

- **Variaciones de algoritmo 2.**

  Estas variaciones para el segundo modelo también fueron mayormente estructurales (capas), agregando Dropout en la salida del bloque Inception y capas Densas de activación ReLu. También se varió el optimizador entre Adam y SGD.

  En base a los resultados obtenidos, se puede notar que la primera variación fue la que obtuvo mejores resultados, donde se aplicó el optimizador SGD con valor de 0.1 además de las cuatro capas principales más la capa de salida. Su valor de Accuracy

Tab. 3.2: Variaciones realizadas al algoritmo 2

Variación	Capas	Optimizador	Loss	Accuracy	ROC-AUC
1	4 capas + salida	SGD(0.1)	0.033	0.987	0.922
2	4 capas + salida	adam	0.2093	0.9239	0.889
3	4 capas + Densa + Dropout + salida	SGD(0.1)	0.0241	0.9908	0.911
4	4 capas + Densa + Dropout + salida	adam	0.2333	0.9507	0.799

- Para los nombres y links de archivos ver anexo 2.

es levemente inferior al de la variación tres, pero su ROC-AUC es mejor, por lo que clasifica de mejor manera las emociones individualmente.

Además, las variaciones se probaron con dos vídeos de YouTube que son detallados más adelante, en el capítulo de validación

Finalmente, luego de las variaciones, nos quedamos con la primera variación del algoritmo, dado que sus valores de accuracy y ROC-AUC fueron mejores. Además las pruebas realizadas en vídeo fueron bastante más consistentes.

De igual manera como ocurrió con el dataset, se descartó probar más modelos y variaciones por la amplitud de la solución a implementar y el tiempo empleado.

3.3.2. UI

Para la creación de la interfaz, se diseñó junto a la FALP un sistema capaz de trabajar en paralelo a la teleconsulta. Es por esto que se utiliza Fast Api, además de su simplicidad y conocimientos previos en Python por parte del equipo de desarrollo. Destacar que la interfaz se desarrolla como prueba de concepto, considerando solamente una arquitectura cliente-servidor, donde el servidor asume el rol de médico y por ende el cliente es el paciente. El paciente envía un flujo de vídeo hacia el servidor (computador del médico) donde se procesa el byte-array enviado y se muestra con la integración en la interfaz del médico.

La interfaz fue implementada a través de un script para servidor WebSocket el cual se encuentra constantemente esperando la recepción de un mensaje *byte-array* transformable al formato de OpenCV, para ser procesado por los módulos y mostrado al médico en forma de vídeo con información.

Listing 3.1: Versión simplificada del Websocket

```
app = FastAPI()

@app.websocket("/ws")
async def websocket_endpoint(websocket: WebSocket):
    await websocket.accept()
    try:
        while True:
            contents = await websocket.receive_bytes()
            arr = np.frombuffer(contents, np.uint8)
            frame = cv2.imdecode(arr, cv2.IMREAD_UNCHANGED)
            cv2.imshow('frame', frame)
            cv2.waitKey(1)
    except WebSocketDisconnect:
        cv2.destroyWindow("frame")
        print("Client disconnected")
```

El bloque de código superior es la versión simplificada del WebSocket creado. Este recibe igualmente un *byteArray*, pero lo muestra sin procesar nada.

La versión del cliente se puede apreciar en más detalle en el anexo 3, mientras que el anexo 6 muestra la integración con el módulo de bpm.

4. VALIDACIÓN

Antes de comprobar la eficacia del modelo 2, se utilizó Open CV para procesar los frames del vídeo y se cargó el detector de rostro (o extractor de características) *"haarcascade_frontalface_default.xml"*. Adicionalmente, se utiliza la misma función *inception_v3.preprocess_input* de Keras para InceptionV3, donde se busca evitar el desvanecimiento del gradiente.

Para la primera prueba, se realizó una comprobación de efectividad utilizando dos vídeos de Youtube editados con las emociones de felicidad, enojo, tristeza, disgusto, asombro, neutro y miedo. Esta primera prueba fue meramente visual, dado que los vídeos venían etiquetados con la emoción respectiva lista para comparar con la emoción detectada por el modelo. En la figura 4.1 se ejemplifica uno de los vídeos, mientras que en la siguiente se aprecia cómo detecta la emoción.

Fig. 4.1: Ejemplo de vídeo 1

Fig. 4.2: Detección de emoción sorpresa en vídeo 1

Los siguientes resultados fueron obtenidos con una inspección visual de los vídeos con el modelo activo.

En ella se muestra el *logit score* en escala (0 - 1) de la emoción detectada versus la *ground truth*. Esto se hizo para ambos vídeos. Se pudo apreciar que la primera emoción del vídeo (tristeza) era generalmente confundida con neutralidad o miedo en su mayoría. Esta falla fue asociada a la mala interpretación de la tristeza por parte de los actores. La siguiente emoción es sorpresa, la que fue detectada de buena manera con al rededor de un 80 % de precisión. La penúltima emoción del vídeo era *Emocionado*, la cual interpretamos como feliz al ser sinónimo, así que la detección de esta emoción fue de al rededor de un 90 % en los tres segundos que nos presentó el vídeo. Finalmente, se encuentra Enojo, la cual fue detectada prácticamente en un 100 % de los frames del vídeo en los que esta emoción estaba presente, exceptuando en aquellos donde el extractor de características no pudo reconocer la cara del sujeto.

En el segundo vídeo, todas las emociones parten en neutro, estado que fue detectado a la

perfección, luego de un par de segundos, la persona comienza a manifestar la emoción. En primer caso está sorpresa, la que fue detectada como miedo por el modelo. La siguiente emoción es disgusto, que fue detectada a la perfección. Seguida a esta emoción, se encuentra la felicidad que de igual manera fue detectada en un 100 % de los frames donde estaba presente. Después viene tristeza la cual nuevamente tuvo problemas en ser detectada pues fue predicha como miedo o enojo, pero nunca tristeza. En penúltimo lugar está enojo, la cual fue detectada con precisión del 100 %. Finalmente, la emoción de miedo fue detectada con al rededor de un 80 % de precisión, dado que fue confundida con enojo.

Tab. 4.1: Tabla de Aciertos/Totales

	Enojo	**Disgusto**	**Miedo**	**Felicidad**	**Tristeza**	**Sorpresa**	**Total**	**Porcentaje**
Vídeo 1	5/5	4/14	31/48	11/11	0/55	0/24	51/157	32.48 %
Vídeo 2	57/58	N/A	N/A	21/27	19/167	11/40	108/292	36.98 %
Total	62/63	4/14	31/48	32/38	19/222	11/64		
Porcentaje	98.41 %	28.57 %	64.58 %	84.21 %	8.55 %	17.18 %		

Se omite emoción Neutro. Aciertos/Totales en cantidad de Frames.

En la tabla superior, se aprecia con bastante más claridad los frames con la emoción detectada. Se omite Neutro a propósito dado que se detectó con una eficacia del 100 % durante las pruebas realizadas. Igualmente, la poca detección de tristeza puede ser por la mala interpretación de los sujetos de prueba, ya que, visualmente, se pudo notar que no eran tan buenos actores. Por otro lado, disgusto, pese a tener poca cantidad de datos se detectó en mayor cantidad que sorpresa, aproximadamente un 10 % más. Los porcentajes más altos de detección fueron Enojo y Felicidad, lo que significa que probablemente el Dataset FER-2013 está sesgado.

En conclusión, según las pruebas realizadas el modelo tiene una efectividad al rededor del 34 % en pruebas realizadas en vídeo, pero igualmente en la detección a nivel humano, es bastante acertado e interpretable en la emoción representada.

5. CONCLUSIONES

En esta memoria realizada en el contexto del programa de Memorias Multidisciplinarias, se exploró la posibilidad de utilizar la detección de emociones a través de una cámara junto a modelos de machine learning, para mejorar el trato de personas en las consultas de telemedicina oncológica. Para lograr esto se implementó el modelo InceptionV3, el cual demostró ser capaz de detectar las emociones con una precisión superior al 60 % en promedio. A pesar de que inicialmente fue entrenado el sistema para reconocer seis emociones diferentes, se decidió descartar la emoción de disgusto debido a su baja detectabilidad. Este hallazgo ha permitido focalizar el trabajo en las emociones que el sistema puede identificar de manera más efectiva, mejorando así la eficiencia general del modelo.

Además se desarrolló una interfaz de usuario que cumple con los requisitos de la prueba de concepto propuesta por la FALP, lo que ha permitido una integración más fluida del sistema propuesto en el entorno clínico. También, se ha logrado la integración de módulos adicionales al sistema, como la detección del ritmo cardiaco y la caracterización de la atención. Estos módulos han aumentado la funcionalidad del sistema, permitiendo ofrecer un producto mínimo viable (MVP) que puede ser utilizado en un entorno clínico.

A pesar de los logros obtenidos, reconocemos que nuestro sistema actual tiene varias limitaciones que deben abordarse. Una de estas limitaciones es la validación incorrecta de las emociones en una base de frame a frame. Esta limitación puede llevar a una interpretación incorrecta de las emociones del paciente, lo que puede afectar la eficacia de las consultas de telemedicina. Además, creemos que hay un potencial significativo para mejorar la precisión de nuestro sistema mediante la exploración de otros backbones de deep learning existentes en la literatura.

Trabajo futuro para este proyecto es una verificación más amplia del rendimiento de los

backbone de deep learning existentes en la literatura con el fin de verificar la viabilidad de su aplicación en tiempo real o para aumentar aún más la precisión, dado que la versión actualmente implementada presenta una incorrecta validación de emociones frame a frame, lo que se puede interpretar como un mal modelo de detección. Así mismo, queda pendiente realizar la implementación del diseño en paralelo a la teleconsulta utilizando una captura de pantalla u otro medio de captura.

Bibliografía

[1] Abhijeet3922. 2019, Enero 7. *FaceEmotion_ID*. https://github.com/abhijeet3922/FaceEmotion_ID.

[2] Ali Altaher, Zahra Salekshahrezaee, Azadeh Abdollah Zadeh, Hoda Rafieipour, Ahmed Salem Abdulmajeed Altaher. 2021, Mayo. *Using Multi-inception CNN for Face Emotion Recognition* https://www.researchgate.net/publication/351345382_Using_Multi-inception_CNN_for_Face_Emotion_Recognition.

[3] Ali I. Siam, Naglaa F. Soliman, Abeer D. Algarni, Fathi E. Abd El-Samie, Ahmed Sedik. 2022 ,02 de Febrero. *Deploying Machine Learning Techniques for Human Emotion Detection*. https://www.hindawi.com/journals/cin/2022/8032673/.

[4] Octavio Arriaga, Paul G. Ploger, Matias Valdenegro. 2017, 20 de Octubre. *Real-time Convolutional Neural Networks for Emotion and Gender Classification*. https://arxiv.org/pdf/1710.07557.pdf.

[5] Nivrito, A., Wahed, M.R. 2016, 18 de Agosto. Comparative analysis between Inception-v3 and other learning systems using facial expressions detection.

[6] Koustabh Das. 2022, 20 de Marzo. *Emotion Detection*. https://www.kaggle.com/code/koustabh98das/emotion-detection/notebook#Building-CNN.

[7] Rohit Pathar, Abhishek Adivarekar, Arti Mishra, Anushree Deshmukh. 2019, 21 de Junio. *Human Emotion Recognition using Convolutional Neural Network in Real Time*. https://ieeexplore.ieee.org/document/8741491.

[8] Sanskar Hasija. 2022, 05 de Enero. *EMOTION DETECTION*. https://github.com/sanskar-hasija/kaggle/blob/118315d8284452f00dc95af6d23f02cc7ab6c01a/emotion-detection.ipynb

[9] Rajasekaran Thangaraj, P Pandiyan, T Pavithra, V.K Manavalasundaram, R Sivaramakrishnan, Vishnu Kumar Kaliappan. 2022, 18 de Enero. *Deep Learning based Real-Time Face Detection and Gender Classification using OpenCV and Inception v3* https://ieeexplore.ieee.org/document/9675635.

[10] Dumitru, Ian Goodfellow, Will Cukierski, Yoshua Bengio. 2013. *FER2013 (Facial Expression Recognition 2013 Dataset)* https://www.kaggle.com/competitions/challenges-in-representation-learning-facial-expression-recognition-ch data.

[11] Lundqvist, D., Flykt, A., & Öhman, A. (1998). The Karolinska Directed Emotional Faces - KDEF, CD ROM from Department of Clinical Neuroscience, Psychology section, Karolinska Institutet, ISBN 91-630-7164-9

[12] P. Lucey, J. F. Cohn, T. Kanade, J. Saragih, Z. Ambadar and I. Matthews, "The Extended Cohn-Kanade Dataset (CK+): A complete dataset for action unit and emotion-specified expression,"2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops, San Francisco, CA, USA, 2010, pp. 94-101, doi: 10.1109/CVPRW.2010.5543262.

[13] James Sandy. 2021, 4 de Enero https://www.section.io/engineering-education/choosing-between-django-flask-and-fastapi/#:~:text=Education%20Django%20is%20more%20strenuous,has%20the%20fewest%20online%20resources.

[14] Django Software Foundation. 2005. *Django*. https://www.djangoproject.com/.

[15] Armin Ronacher. 2010, 1 de Abril. *Flask*. https://flask.palletsprojects.com/en/2.2.x/.

[16] Sebastián Ramírez. 2018, 5 de Diciembre. *FastApi*. https://fastapi.tiangolo.com/.

6. ANEXOS

6.1.

Nombres y links de archivos para variaciones de algoritmo 1:

- Variación 1: Test1.ipynb

  `https://drive.google.com/file/d/1y1tQDZhm0i-aDM3WH4cqv3HkSgkt8Lwv/view?usp=sharing`

- Variación 2: Test1_Sin_Validation.ipynb

  `https://drive.google.com/file/d/10OydqcZ6Fh6-9Sz1N2HQEialKmIGwBWU/view?usp=sharing`

- Variación 3: Test1_con_validation.ipynb

  `https://drive.google.com/file/d/1t8czjd_PwuOYo5EjN6Wmv761TZhgcvid/view?usp=sharing`

6.2.

Nombres y links de archivos para variaciones de algoritmo 2:

- Variación 1: Test2 copy 2 FUNCIONA VIDEO sin resize.ipynb

  `https://drive.google.com/file/d/1JLT66G0llP5sf5saHScyjbdY8nLD0Q4M/view?usp=sharing`

- Variación 2: Test_2_estandar.ipynb

  `https://drive.google.com/file/d/1eutg4RfBeMCo1-AdKAtHDnisU4RAdV9/view?usp=sharing`

- Variación 3: Test2.ipynb

  https://drive.google.com/file/d/1B74M6SrHizloDQwWoZIjJ-F59h9NR_AC/view?usp=sharing

- Variación 4: Test_2 _estandar copy.ipynb

  https://drive.google.com/file/d/1Dw6aWrG_n_Nr7s36xjkQ7-9T6o9UIGTb/view?usp=sharing

6.3.

Listing 6.1: Código de conexión del cliente

```
import websockets
import asyncio
import cv2

camera = cv2.VideoCapture(0, cv2.CAP_DSHOW)

async def main():
    # Connect to the server
    async with websockets.connect('ws://localhost:8080/ws') as ws:
    # async with websockets.connect('ws://localhost:8080/ws') as ws:
        while True:
            success, frame = camera.read()
            if not success:
                break
            else:
                ret, buffer = cv2.imencode('.png', frame)
                await ws.send(buffer.tobytes())
                cv2.imshow('frame', frame)
                cv2.waitKey(1)
# Start the connection
asyncio.run(main())
```

6.4.

Link a vídeo demostrando algoritmo final escogido

`https://drive.google.com/file/d/1BG8eLF3CTsme02riHNuUvtMrTZcAU3KU/view?usp=sharing`

6.5.

Link a vídeo de prueba de interfaz solo detección de emociones

`https://drive.google.com/file/d/13mwfuPU94l4R6MTCqtuI_Erp8qolxBtf/view?usp=sharing`

6.6.

Link a vídeo de interfaz con modulo bpm integrado

`https://drive.google.com/file/d/12bMV5-4SL4hRnU7OTkBhjZsnLKHAVWNW/view?usp=sharing`