Posgrado en Ciencias y Tecnologías de la Información

Estimación de características óptimas para la generación de deepfakes.

Idónea Comunicación de Resultados

para obtener el grado de:

Maestra en Ciencias y Tecnologías de la Información

Presenta:

Paola Polet Beltrán García

Asesores:

Dr. Pedro Lara Velázquez

Dr. Eric Alfredo Rincón García

Sinodales:

Presidenta: Dra. Bibiana Obregon Quintana

Vocal: Dr. Roman Anselmo Mora Gutierrez

Secretario: Dr. Pedro Lara Velázquez

Resumen

En la actualidad, la mayoría de las personas son consumidores de entretenimiento y noticias principalmente por redes sociales y la web. En ellas se puede encontrar fácilmente cualquier tipo de contenido o bien los temas más populares del momento, los cuales en su mayoría no son cuestionados acerca de su veracidad. En estos últimos años se han divulgado muchos videos manipulados con fines de entretenimiento y lucro, en donde usualmente se ven rostros de artistas en películas que no fueron parte del reparto, en eventos dando discursos fuera de lo común, o bien realizando actividades cotidianas de una forma peculiar y que seguramente resultan creíbles por lo bien que están hechos, o porque no se presta la atención suficiente para dudar de ellos. En algunos casos estos materiales fueron creados con una nueva tecnología conocida como deepfake, que se utiliza para poner el rostro de una persona sobre el de otra, lo cual permite fácilmente hacer ediciones y montajes, utilizando una gran cantidad de imágenes de las personas que se quiera falsificar y con la ayuda de softwares creados con Inteligencia Artificial (IA). Esta tecnología puede mostrarse como divertida, sin embargo, este tipo de contenido puede generar una gran cantidad de desinformación y daños morales. Aunque el DEEPFAKE es una técnica muy reciente, existen trabajos muy bien ejecutados, no obstante, también se pueden encontrar resultados menos creíbles. En este trabajo de investigación se propone estimar los factores que influyen en la realización de este tipo de contenido, para así poder determinar las mejores condiciones y obtener resultados idóneos, sin la necesidad de hacer uso de herramientas de diseño o aplicaciones para mejorar errores. Para ello se hicieron pruebas con características relativas de los videos originales como la iluminación, tono de piel, tipo de encuadre, afinidad de los rostros y movimientos del conjunto de datos. Se estudiaron los tipos de iluminación, se hicieron modificaciones sobre el tono de piel en los videos originales, se generó un algoritmo que permitiera identificar la afinidad entre dos rostros, además se creó una matriz de similitud en donde se pudiera identificar más

rápido el parecido entre dos celebridades, se crearon *deepfakes* con diferentes ángulos y movimientos del rostro, entre otras cosas. También se revisaron y modificaron algunos hiperparámetros de una red neuronal que permite realizar *deepfakes* como el optimizador y el número de capas que utiliza la red, todo lo mencionado anteriormente con el fin de observar el desempeño en los resultados. Esta investigación tiene un aporte más sobre la línea de detección de *deepfakes* en los cuales los puntos que se mencionarán en la fase de pruebas pueden ser útiles como base para identificar falsedades en este nuevo material.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

En los últimos años se han divulgado videos cuyo contenido comúnmente muestra celebridades dando discursos, entrevistas o realizando actividades las cuales ellos no reconocen como propias, y que fueron creados de una manera tan creíble que los espectadores no dudan de su veracidad. Esta tecnología es conocida como DEEPFAKE y consiste en poner la cara de una persona sobre la de otra de forma que el resultado luzca realista. Los *deepfakes* surgen en el año 2017 y fueron creados inicialmente por un usuario de la plataforma Reddit con el seudónimo *"deepfake"*, el cual aseguraba haber creado un algoritmo que permitía intercambiar los rostros de celebridades en videos pornográficos. El término proviene de la combinación de las palabras *deep* que hace referencia a la técnica que se utiliza para la realización de estos videos, *deep learning* o *aprendizaje profundo*, y *fake* que significa falso [1].

Esta técnica se ha popularizado debido a la cantidad de contenido que ha sido generado y compartido en redes sociales, el cine, la televisión y otros medios. El impacto fue tan extenso que una gran cantidad de personas interesadas han desarrollado programas y aplicaciones similares que permiten crear *deepfakes* fácilmente. Estos sistemas basados en Inteligencia Artificial (IA) utilizan algunos tipos de redes neuronales como autoencoders o las redes generativas antagónicas (GAN). Cabe mencionar que los recursos y el tiempo para generarlos son accesibles, lo que facilitan su trabajo.

Esta nueva tecnología trae consigo un gran avance en el medio del entretenimiento, la investigación, campañas publicitarias, entre otras áreas, beneficiando principalmente a la industria cinematográfica, en donde se pueden hacer cambios en los diálogos o en las escenas sin la necesidad de volver a grabar, también se puede revivir a artistas fallecidos

y rejuvenecer o envejecer la imagen de una persona. De igual forma, ha favorecido a la industria de videojuegos, la fotografía, el comercio electrónico, el modelado 3D, entre otras [1]. Para generar este tipo de contenido se debe aplicar una normativa la cual avale el uso de la identidad de la persona o bien que los fines de generar estos *deepfakes* no sean malintencionados y no afecte la imagen de la persona en cuestión. Entre algunos ejemplos recientes presentados en la televisión mexicana se encuentra la recreación de la imagen del cantante Juan Gabriel junto con el comediante Cantinflas y Blue Demon en un comercial patrocinado por una compañía de cervezas[1]. En este anuncio se invita a la audiencia a disfrutar de un concierto tributo con motivo especial del Día de Muertos, con la participación de otros artistas y el apoyo de la tecnología DEEPFAKE encargada por la empresa *Reality Kraft*. En otro comercial de la cadena de supermercado Soriana[2] reviven también la imagen de Cantinflas. Sin embargo, existen otras aportaciones populares, como lo fue en España en donde utilizaron la imagen de la actriz Lola Flores para una campaña de Cruzcampo[3], o el caso en el que apareció un cameo de la rejuvenecida princesa Lea (Carrie Fisher) para la película de *Star Wars Rogue One*[4] en donde la actriz que llevaba el papel cumplía los 60 años de edad. En Youtube se puede encontrar un video donde se substituyen los rostros de los actores Robert Downey Jr. y Tom Holland en una escena de la película Volver al Futuro[5], o también se puede encontrar al actor Steve Buscemi en el cuerpo de la actriz Jennifer Lawrence en su discurso de los Golden Globes Awards[6] y una gran cantidad de ejemplos más.

Sin embargo, además de estos ejemplos de DEEPFAKE, ya sean creados de manera profesional o como entretenimiento, tienen una gran cantidad de desventajas principalmente la expansión de *fake news*, en donde con ayuda de las redes sociales hoy en día su divulgación es más rápida y masiva. Otras desventajas son las falsificaciones de videos en política, acoso, engaños, fraudes financieros, creación de perfiles falsos e invasión de privacidad cuando no existe el consentimiento de las personas involucradas. Algunos ejemplos incluyen el comunicado falso de navidad dado por la reina Isabel[7] o bien el video donde el

[1] `https://www.youtube.com/watch?v=VsO_UT3SLkE&ab_channel=VictoriaMxico`
[2] `https://www.youtube.com/watch?v=EpzUryzTZGc&ab_channel=Soriana`
[3] `https://www.youtube.com/watch?v=Yewm6TfLZ3Q&ab_channel=cruzcampoTV`
[4] `https://www.youtube.com/watch?v=6Yj31YCa3Xw&ab_channel=theryaney`
[5] `https://www.youtube.com/watch?v=veeknp4Bbpo&ab_channel=DelComicAlCine`
[6] `https://www.youtube.com/watch?v=0dKMXe_pbKQ&ab_channel=KnackerBags`
[7] `https://www.youtube.com/watch?v=xNOqqTa5EfU&ab_channel=Comercialeschingones`

ex presidente de los Estados Unidos, Barack Obama, insulta al presidente Donald Trump[8], incluyendo una recomendación de lo cuidadoso que se debe ser hoy en día con lo que se ve.

A pesar de todos los ejemplos mencionados anteriormente, esta tecnología es muy reciente y desconocida. Se han desarrollado varios softwares que permiten generar *deepfakes* de forma simple, aunque los resultados no siempre suelen ser convincentes, por lo que a veces un video puede requerir el uso adicional de herramientas que permitan hacer modificaciones en el rostro resultante y así perfeccionar el producto final. Por otro lado, se ha demostrado en [2] [3] que poca gente conoce sobre la técnica DEEPFAKE y que aunque se les advierta que existe este tipo de manipulaciones en videos, es todavía muy difícil que sepan identificar un video falso de uno original aun cuando aseguren poder hacerlo. Por todo lo anterior, la motivación de este proyecto se basa en detectar la influencia que tienen ciertas características en los videos para obtener *deepfakes* creíbles. Se espera que este trabajo ayude y motive la necesidad de analizar cuidadosamente y ser critico sobre el contenido que se encuentre en la red, ya que ahora no solo bastará ver para determinar si lo que se presenta es real o no.

Dentro de este trabajo se mencionará mucho el término **deepfake**, el cual, puede referirse a la técnica para realizar el intercambio de rostros que estará denotado como DEEPFAKE, o bien puede referirse a los videos resultantes, los cuales ya han sufrido este cambio de caras. A estos resultados se les denotará como *deepfake*.

1.1. Objetivos

Objetivo general

Estimar las características idóneas para la realización de videos *deepfakes*, para así obtener resultados más realistas.

Objetivos particulares

- Realizar un estudio del estado del arte sobre los conceptos más importantes sobre redes neuronales básicas, redes convolucionales y autoencoders.
- Revisar la herramienta *"DeepFaceLab"* para la generación de *deepfakes*.

[8] `https://www.youtube.com/watch?v=cQ54GDm1eL0&ab_channel=BuzzFeedVideo`

- Experimentar sobre un conjunto reducido de videos y estandarizar los parámetros y los pasos que se realizan después del entrenamiento del modelo.
- Determinar algunas características de los videos originales que afecten el *deepfake*.
- Calibrar algunos hiperparámetros de la red neuronal.
- Estimar el mejor desempeño esperado, estimar el costo mínimo de la función *loss*.

1.2. Metodología

Para lograr los objetivos previamente planteados se propuso trabajar con la siguiente metodología.

- Estudiar el funcionamiento de la herramienta *"DeepFaceLab"*.
- Seleccionar un conjunto de videos de corta duración para la realización de pruebas.
- Utilizar un mismo video como origen y destino para cerciorarse del buen funcionamiento del sistema en uno de los casos más sencillos.
- Utilizar videos diferentes de la misma persona como origen y destino para estudiar como influye en el resultado tener diferentes características de fondo pero un mismo rostro.
- Utilizar videos de diferentes personas como origen y destino para examinar el acoplamiento de dos rostros distintos.
- Utilizar diferentes e iguales conjuntos de datos como origen y destino pero con variaciones por ejemplo de iluminación, tiempo de entrenamiento, entre otras.
- Identificar los factores que afectan y/o benefician en la generación de *deepfakes*.
- Seleccionar y usar videos de diferentes personas considerando las medidas antropométricas de sus rostros.
- Determinar las ventajas, en caso de existir, al seleccionar como fuentes de origen y destino a rostros con medidas antropométricas semejantes y/o con las características favorables que se hayan encontrado.

- Estudiar y realizar la calibración de algunos hiperparámetros como el número de capas, número de *epochs* y optimizador.

1.3. Estructura de la idónea comunicación de resultados

En esta primera sección se introduce al lector al proyecto de investigación, además de mostrar los objetivos del mismo y la metodología a seguir para alcanzar dichas metas. En el Capítulo 2 se exponen trabajos relacionados los cuales aportan información sobre la tecnología DEEPFAKE, así como los sistemas que se podían utilizar para realizarlos, sus características y modelos. En este capítulo también se habla de las redes neuronales autoencoders y las formas en las que son aplicadas para la realización de *deepfakes*. En el Capítulo 3 se da una explicación del sistema *DeepFaceLab*, sus características y su funcionamiento. En el Capítulo 4 se presentan los experimentos realizados con cada una de las características probadas, así como los resultados de cada uno de estos ensayos de prueba. Finalmente en el Capítulo 5 se presentan las conclusiones de las pruebas realizadas.

Capítulo 2

Estado del arte

Antes de mencionar los artículos revisados en la literatura especializada, es importante una breve introducción a algunos conceptos que son de importancia para este trabajo, por lo cual se dedicarán unas secciones para describir qué son y cómo funcionan las redes neuronales y las redes convolucionales. Una revisión más detallada se incluye en los Apéndices A y B. Los conceptos de autoencoders se encuentra dentro de este capítulo.

2.1. Modelos y sistemas *deepfake*

En esta sección se describirán los modelos, sistemas y técnicas DEEPFAKE encontradas en la literatura. En [1] se presentan las técnicas de manipulación que existen de imágenes y videos, así como los criterios y métodos que se toman en consideración para descubrirlas. Para cada grupo de modificación los autores proporcionan detalles sobre estas técnicas, bases de datos públicas existentes y puntos de referencia clave para la evaluación tecnológica de métodos de falsificaciones. Este artículo fue utilizado como referencia para determinar las características que deben considerarse al crear *deepfakes*, así como para detectar puntos claves en estas alteraciones.

Las técnicas de manipulación que mencionan en [1] se clasifican en cuatro grupos.

- Para imágenes

  Síntesis facial completa. Consiste en la creación de rostros de personas inexistentes realizadas generalmente a través de las redes generativas adversarias (GAN).

  Manipulación de atributos. Conocida también como edición facial o retoque facial, consiste en modificar algunos atributos del rostro como color del cabello o

de la piel, el género, la edad, añadir gafas, bigote, entre otros elementos. De igual forma este proceso se realiza comúnmente a través de GAN.

- Para video

Intercambio de rostros. Consiste en colocar el rostro de una persona en el de otra.

Intercambio de expresión. Permite utilizar los movimientos de la cara en un video para guiar los movimientos de otro rostro en una imagen u otro video.

En [4] los autores entran en un campo más amplio considerando DEEPFAKE de audio, video e imágenes añadiendo otras técnicas para manipulaciones como: sincronización de labios, en la cual, los movimientos de un individuo objetivo se transforman para hacerlos consistentes con alguna grabación de audio. Síntesis facial y manipulación de atributos, en donde se generan imágenes faciales fotorrealistas y editan atributos faciales y finalmente, usan la técnica de DEEPFAKE para falsificaciones de audio o clonación de voz.

La Tabla 2.1 muestra un resumen de las técnicas más relevantes de manipulación, los programas que permiten su realización y el tipo en el que están clasificados, es decir, si son de código abierto, públicos, se pueden analizar o hacer modificaciones según las necesidades del usuario. Entre estos programas se encuentran implementaciones gratuitas, aplicaciones móviles, aplicaciones web comerciales y softwares de escritorio comercial.

Tabla 2.1: Sistemas DEEKFAKE más comunes en la actualidad.

Intercambio de rostro	
Tipo	Sistema
Código abierto	FaceSwap, DeepFaceLab, FaceSwapGan, FGAN, SimSwap
Código cerrado	ZAO, Reflect, Impressions, FaceShifter, FaceApp
Intercambio de expresión	
Tipo	Sistema
Código abierto	Face2Face, FaceIT3, FSGAN
Código cerrado	Imitator
Manipulación de atributos	
Código abierto	AttGAN
Código cerrado	FaceApp, Adobe, Rosebud.
Clonación de voz	
Código abierto	SV2TTS
Código cerrado	Overdub, Respeecher, ResembleAI, Voicery, VoiceApp.

En [5] presentan *FSGAN*, un algoritmo que permite realizar intercambio y recreación de rostro. La recreación de rostro se refiere a utilizar los movimientos faciales y las expresiones de una cara para guiar los movimientos de otra, es decir intercambio de expresión. Este modelo se puede aplicar a pares de caras sin requerir un entrenamiento de ellas. Su arquitectura está basada en redes neuronales especializadas para cada tarea, una red para la recreación, la segunda para la segmentación de caras, la tercera es un generador de pintura y una más que permite mezclar la información adquirida en la imagen final. Los autores muestran los resultados obtenidos por su algoritmo y los comparan con los producidos por otras técnicas.

En [6] se presenta *SimSwap*, un marco eficiente que es capaz de transferir la identidad de una cara fuente a una cara objetivo al tiempo que conserva los atributos de esta, como: expresión, dirección de la mirada, postura y condición de iluminación. Los autores presentan un Módulo de Inyección ID (IIM) que realiza modificaciones en las características de la imagen destino incrustando la información de identidad de la cara de origen, maneja

varias identidades incluso si las condiciones de destino son difíciles por ejemplo con expresiones exageradas. Su arquitectura utiliza autoencoders y se compone en tres partes, un codificador, el IIM y la parte del decodificador. El codificador extrae funciones de la imagen destino, el IIM transfiere la información de identidad y el decodificador restaura las características modificadas a la imagen del resultado. Exhiben los resultados de sus pruebas en una matriz de caras generadas, donde las imágenes destino fueron seleccionadas de escenas de películas y las imágenes de origen fueron descargadas de Internet. De igual forma que el artículo anterior, realizan varias comparaciones con *Deepfakes, FaceShifter y FSGAN*, con las cuales muestran que es menos probable que los resultados de *Simswap* se vean afectados por los atributos de las imágenes de la fuente de entrada, además de lograr un rendimiento de identidad competitivo al tiempo que preservan mejores atributos como postura y expresión.

En [7] proponen un marco de dos etapas llamado *FaceShifter* para el intercambio de rostros con reconocimiento de oclusión y alta fidelidad. La arquitectura está compuesta por autoencoders y se divide en dos etapas. En la primer etapa se utiliza una red de integración adaptativa (AEI-net), la cual se compone de un codificador de identidad, un codificador de atributos multinivel y un generador con capas de desnormalización atencional adaptativa (AAD). Los autores aluden que desde esta etapa genera la cara intercambiada, explotando e integrando los atributos de destino de manera exhaustiva y adaptativa, para generar un resultado de intercambio de rostros de alta fidelidad. En la segunda etapa utilizan una red de reconocimiento de errores heurísticos (Hear-net), para manejar las oclusiones faciales y refinar el resultado. Este artículo realiza comparaciones cualitativas con *FaceSwap, Nirkin et al., Deepfakes*, e *IP-GAN*. En los resultados *FaceShifter* conserva mejor las formas faciales de las identidades de origen y también es más leal a los atributos de destino como lo son las iluminaciones, resoluciones de imágenes, entre otros. Por otro lado se realizaron comparaciones con *FSGAN* y ambos conservan las oclusiones comunes muy bien, sin embargo *Faceshifter* tiene una ventaja sobre la calidad del rostro y la fidelidad de las entradas.

Finalmente, en [8] muestran *DeepFaceLab* (DFL) un sistema DEEPFAKE de código abierto el cual ha sido muy popular entre el público, que tiene como objetivo lograr resultados de intercambio de rostros fotorrealistas sin la necesidad de realizar grandes ajustes. El funcionamiento de *DeepFaceLab* está basado en tres fases. La primera fase

llamada extracción en donde se realiza la detección, alineación y segmentación de rostros, la segunda fase es el entrenamiento, donde se ofrecen dos estructuras compuestas por autoencoders, haciendo notar que esta fase juega el papel más importante. Por último, la conversión en donde los usuarios pueden intercambiar la cara fuente a la cara destino. Para la realización de estas fases *DeepFaceLab* hace uso de otras herramientas para la detección facial, la extracción de puntos de referencia faciales y la segmentación de rostros. En este trabajo los autores comparan el rendimiento de *DeepFaceLab* (DFL) y presentan los resultados cualitativos obtenidos con *DeepFakes, Nirkin et al., Face2Face, FSGAN, FaceShifter*, de forma que se encontró que DFL tiene un rendimiento competitivo entre ellos y es más hábil para retener la pose y la expresión.

2.2. Autoencoders

Como se mencionó en la sección anterior, algunos de los sistemas DEEPFAKE basan su arquitectura en autoencoders, por lo que a continuación, se da una explicación de los mismos.

Los autoencoders son un tipo de red neuronal cuyo objetivo principal es codificar el dato de entrada, comprender esa información y descodificarla para obtener como resultado una copia de la entrada lo más similar posible [9]. Existen autoencoders de diferentes tipos y cada uno puede ser modificado o combinado para distintas aplicaciones, entre ellas: la clasificación, agrupamiento, detección de anomalías, sistemas de recomendación, reducción de dimensionalidad y ruido, la creación de subtítulos, reconocimiento y generación de imágenes, entre algunos más.

2.2.1. Arquitectura

Los autoencoders se componen principalmente de tres componentes: el codificador, espacio latente también conocido como cuello de botella y el decodificador. Una representación a esta arquitectura se muestra en la Figura 2.1.

Codificador Acepta los datos de entrada, estos datos suelen ser de una dimensión grande y los comprime en datos de menor dimensión.

Espacio Latente. La cantidad de neuronas en esta capa es mucho menor que las usadas al recibir los datos de entrada. La red debe encontrar una manera de representar

los datos lo mejor que pueda con un menor número de neuronas.

Decodificador La tarea del decodificador es reconstruir los datos de entrada de acuerdo a la información recolectada previamente.

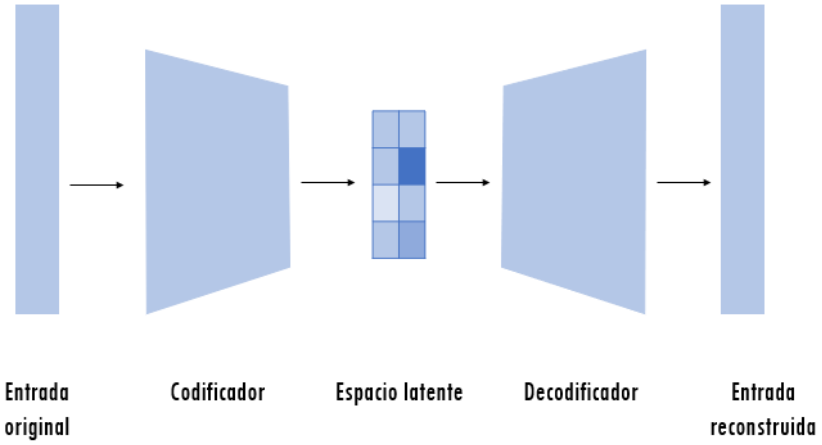


Figura 2.1: Modelo simplificado de un autoencoder.

2.2.2. Autoencoders en *deepfakes*.

Para la generación de *deepfakes* cada una de estas partes realiza ciertas tareas. El codificador recibe una imagen del rostro de dos personas, A y B, y se encarga de extraer los elementos más importantes de la imagen de entrada como: características particulares de la cara, ojos cerrados o abiertos, la posición de la cabeza, expresiones, color de piel, entre otros. Su resultado es una representación de menos dimensiones de esa misma cara, que a veces se denomina cara latente. Ésta representa la cara como una versión comprimida y es también llamada cuello de botella y se usa para que la red pueda aprender características faciales más generales, en lugar de memorizar todos los ejemplos de entrada de personas específicas. La Figura 2.2 muestra una representación de esta cara latente. Finalmente, el decodificador descomprime la información disponible en el espacio latente para reconstruir una imagen lo más fidedigna posible.

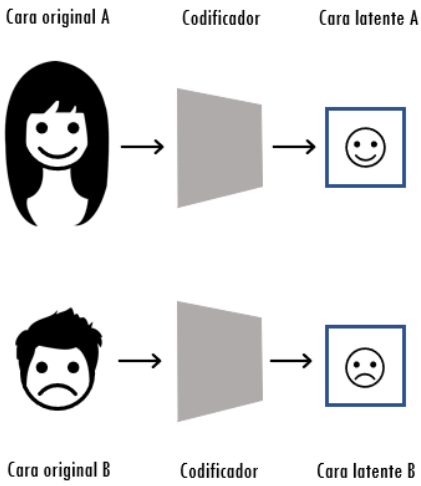


Figura 2.2: Entrada de datos a espacio latente.

Para lograr el intercambio de rostros, el resultado buscado en un *deepfake*, se puede lograr de dos formas. La primera es teniendo dos redes compartiendo el mismo codificador, pero usando dos decodificadores diferentes. Durante la fase de entrenamiento, estas dos redes se tratan por separado. El decodificador A solo se entrena con caras de A; el decodificador B solo con caras de B. Sin embargo, todas las caras latentes son producidas por el mismo codificador. Cuando se completa el proceso de entrenamiento, se puede realizar el intercambio de un rostro latente generado por el sujeto A al decodificador B y este reconstruirá el rostro de la persona B a partir de la información relativa a la persona A [10]. Una representación de este proceso lo muestra la Figura 2.3.

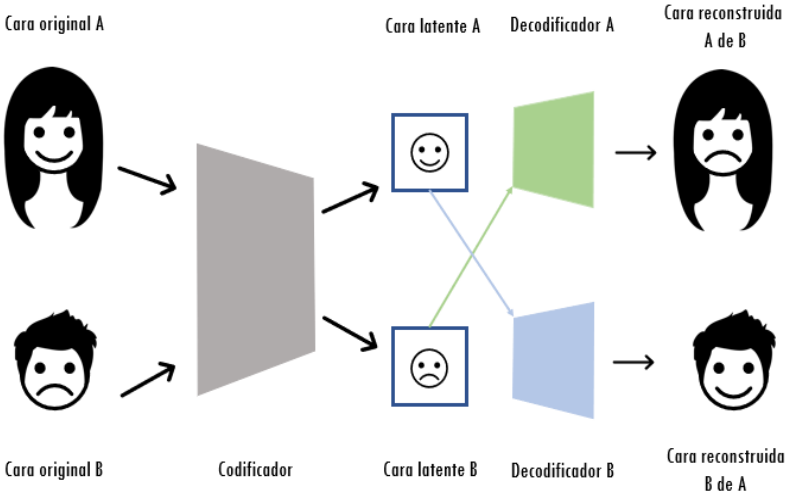


Figura 2.3: Autoencoders con un codificador compartido.

La segunda forma, utilizan dos conjuntos de codificadores y decodificadores. Sin embargo, si ambos decodificadores se entrenan por separado serán incompatibles entre sí y cada decodificador solo podrá decodificar un único tipo de representación latente. La solución está en que los dos codificadores compartan los pesos de la red mientras se utilizan dos decodificadores diferentes. Cuando termina el entrenamiento, la representación latente de la cara que se generó se pasa a la red de decodificadores entrenados en el que se supone se realizará el intercambio. La Figura 2.4 muestra una representación de este proceso. Finalmente, el decodificador reconstruirá un rostro a partir de la información relativa al rostro del sujeto original [10].

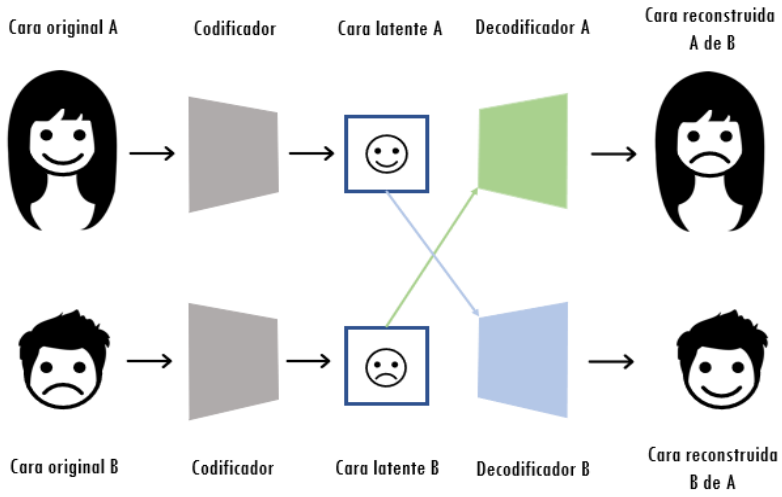


Figura 2.4: Autoencoders con dos decodificadores.

Capítulo 3

DeepFaceLab

DeepFaceLab es un sistema de código abierto para la generación de *deepfakes* creado por Iperov, el cual permite con facilidad la creación de videos *deepfake* sin tener una comprensión amplia sobre el aprendizaje profundo o su implementación [8]. Se encuentra disponible en un servicio basado en la nube conocido por GitHub, que permite que sus usuarios puedan colaborar y hacer cambios en proyectos como *DeepFaceLab*. Este puede modificarse dependiendo de los propósitos de cada usuario logrando resultados favorables sin tener que realizar grandes ajustes.

En este trabajo de investigación se estimaron las características óptimas con ayuda del sistema *DeepFaceLab* para generar *deepfakes*. En este capítulo se describen sus características, las fases principales en las que se divide su funcionamiento y los pasos a seguir en cada una de éstas para la creación de *deepfakes*. Un resumen del procedimiento se muestra en el diagrama de flujo de la Figura 3.17. Las siguientes subsecciones explican más a detalle cada etapa.

3.1. Características de *DeepFaceLab*

Las características principales con las que cuenta *DeepFaceLab* se encuentran enlistadas a continuación [8]:

- Proporciona herramientas de línea de comandos que se pueden ejecutar de la forma que el usuario elija
- Admite configuración de múltiples GPU

- Modelo de enmascaramiento XSeg
- Cuenta con dos modelos de entrenamiento: SAEHD y Quick 96
- Entrenamiento GAN
- Vista previa de entrenamiento
- Fusión interactiva

3.2. Funcionamiento de *DeepFaceLab.*

La canalización o *pipeline* de *DeepFaceLab* se divide en 3 fases principales: la extracción, el entrenamiento y la conversión [8]. *DeepFaceLab* no cuenta con una interfaz gráfica de usuario, en su lugar se tienen varios archivos .bat que realizan los procesos y finalmente generan un *deepfake*. Al realizar la descarga de *DeepFaceLab* además de encontrar estos archivos .bat se encontrarán dos carpetas, la carpeta *internal*, que contiene archivos internos y la carpeta *workspace* que es la carpeta donde se guardará el conjunto de datos que se usarán, modelos, imágenes y videos.

Inicialización del espacio de trabajo

El primer paso consiste en seleccionar el conjunto de datos a utilizar, es decir, el video destino, que va a ser el video donde se realizará el intercambio de caras, y el video fuente, que es el video del que se tomará el rostro de la persona que se desea intercambiar. Éstos se cargarán dentro de la carpeta *workspace* y deberán nombrarse como *data_dst*, y *data_src* respectivamente. Además de los videos se deberán tener las siguientes carpetas vacías: *data_dst*, *data_scr* y *model* (conforme se avance estas carpetas se irán llenando de imágenes y otras carpetas que se mencionarán más adelante). La Figura 3.1 muestra un ejemplo de cómo debería verse el espacio de trabajo después de haber realizado este procedimiento.

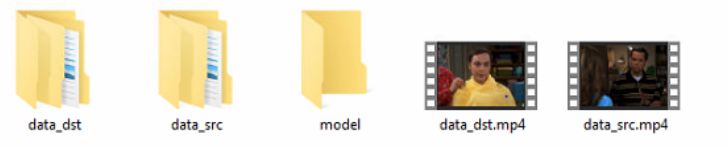


Figura 3.1: Espacio de trabajo.

Después se entra en la etapa de Extracción.

3.2.1. Extracción

Esta fase consiste en la detección, alineación y segmentación del rostro.

Extracción de los fotogramas del video fuente

Se procede a extraer los fotogramas o cuadros del video fuente para convertirlo en secuencia de imágenes (png/jpg) utilizando el archivo **2)extract images from video data_src.bat**. Las imágenes obtenidas serán agregadas a la carpeta *data_src*. Una vez concluido este proceso la carpeta podrá verse como en la Figura 3.2 y su contenido se muestra en la Figura 3.3 la cual tendrá las imágenes extraídas y una carpeta más con nombre *aligned*.

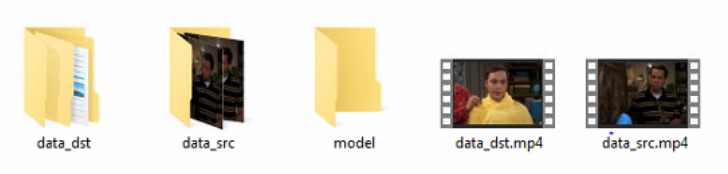


Figura 3.2: Extracción de imágenes - video fuente.

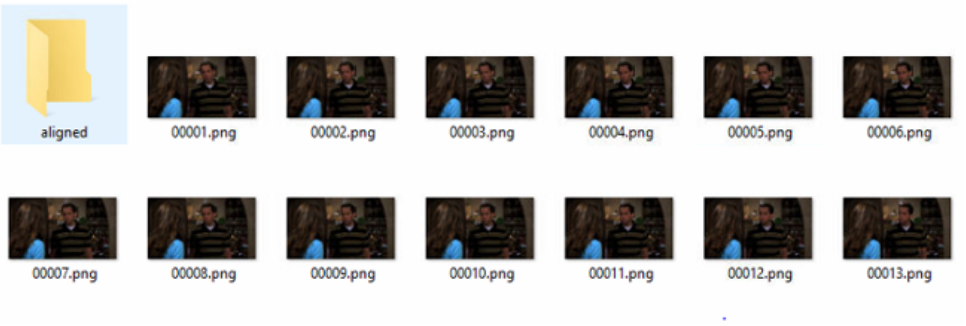


Figura 3.3: Contenido de la carpeta *data_src*.

Extracción de los fotogramas del video destino

Se realizará lo mismo que el paso anterior pero ahora con el video destino y el archivo **3)extract images from video data_dst FULL FPS.bat**. Los archivos se agregan ahora a la carpeta *data_dst* como se ve en la Figura 3.4. Dentro de esta carpeta también se tendrán las imágenes y la carpeta *aligned* como en la Figura 3.5.

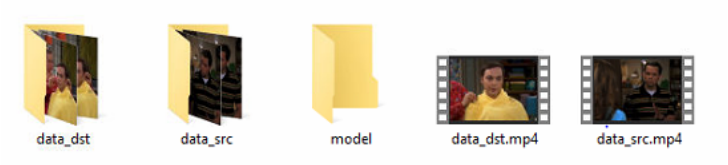


Figura 3.4: Extracción de imágenes - video destino.

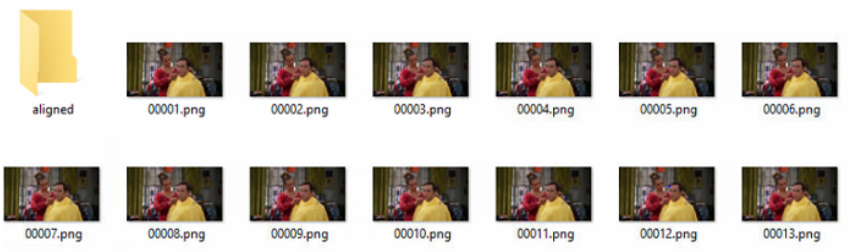


Figura 3.5: Contenido de la carpeta *data_dst*.

Depuración de fotogramas del video fuente y video destino

Una vez obtenidos los fotogramas de cada video, dentro de las carpetas *data_dst* y *data_src* se realizará una depuración de los mismos por si se encuentran rostros de otras personas o bien fotogramas que no incluyan rostros. Se deberá asegurar que todos los fotogramas incluyan imágenes de personas o caras, ya que si se ejecuta el siguiente archivo .bat y se tienen cuadros sin estas características se obtendrá un error en el procedimiento. Esta etapa no se ha automatizado, así que el usuario debe hacerla por si mismo. Las carpetas mencionadas anteriormente deberán verse como la Figura 3.6.

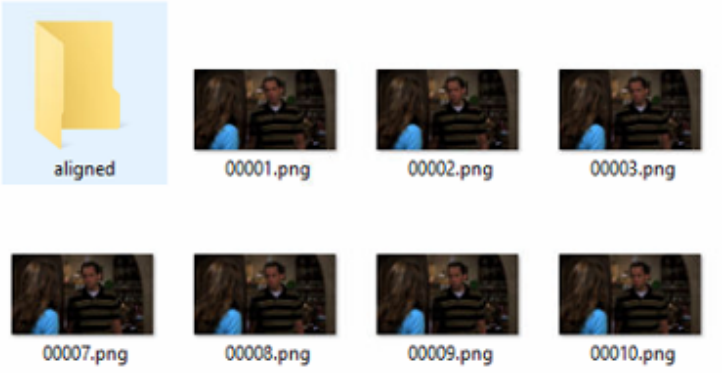


Figura 3.6: Depuración de fotogramas.

Detección de rostros de los fotogramas del video fuente

Posteriormente sigue la parte de detección y alineación de rostros. Esta parte se realiza en conjunto. Se detectará y extraerá el rostro de los cuadros de videos obtenidos ante-

riormente y a continuación se aplicará un algoritmo de puntos de referencia faciales para que pueda mantener estable el rostro a lo largo del tiempo. Este proceso puede realizarse de forma manual con el archivo **4)data_src faceset extract MANUAL.bat** o de forma automática con **4)data_src faceset extract.bat**. La forma automática utiliza el algoritmo S3FD, y se recomienda utilizar esta opción porque es más rápida. Los archivos obtenidos por este paso se encontrarán dentro de la carpeta *data_src/aligned* como se muestra en la Figura 3.7.

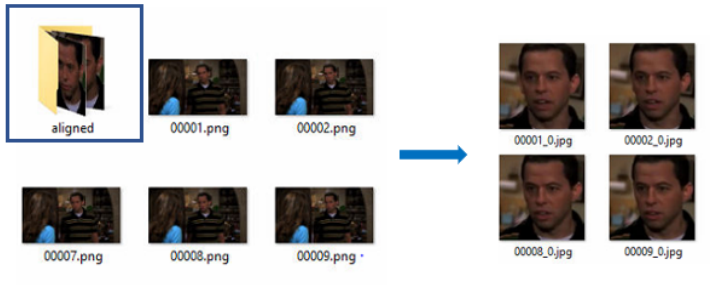


Figura 3.7: Detección y alineación de rostros - video fuente.

Depuración de los rostros detectados del video fuente

Al igual que en el paso 4, se deberá hacer una limpieza de las imágenes, para este procedimiento se tiene el archivo **4.1)data_src view aligned result.bat** el cual abre una "vista", como se aprecia en la Figura 3.8, que incluye las imágenes obtenidas. Se deberá eliminar el conjunto de imágenes que no contengan rostros de personas, también se deberán desechar las imágenes que contengan caras de otras personas diferentes al rostro fuente o destino.

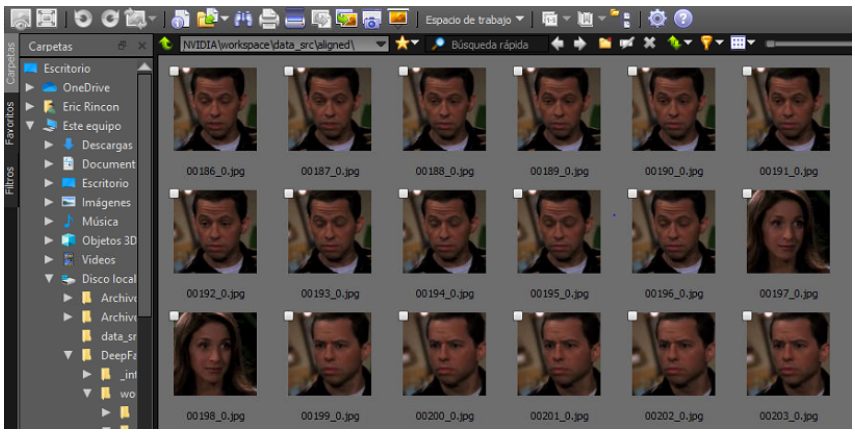


Figura 3.8: Limpieza del conjunto fuente.

Detección de rostros de los fotogramas del video fuente

Ahora, se debe extraer el set de caras encontradas en las imágenes obtenidas anteriormente del video destino con **5)data_dst faceset extract.bat**. Este conjunto de rostros se encontrarán dentro de *data_dst/aligned* como se observa en la Figura 3.9.

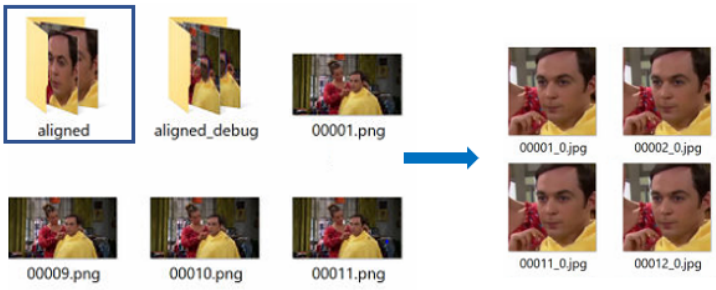


Figura 3.9: Detección y alineación de rostros - video destino.

Depuración de los rostros detectados del video destino

Se realizará la limpieza en las imágenes del rostro destino con **5.1)data_dst view aligned results.bat**, surge lo mismo que en el paso 6, se abrirá una vista para seleccionar las imágenes que no sean rostros. La Figura 3.10 muestra un ejemplo de cómo debería verse el conjunto de imágenes a utilizar.

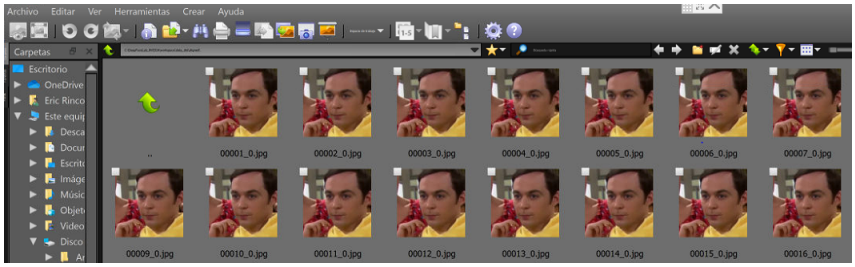


Figura 3.10: Limpieza del conjunto destino.

La parte de segmentación de rostros es opcional y se realiza principalmente cuando se desea eliminar obstrucciones que cubran la cara, o cuando se está trabajando con una cara completa. *DeepFaceLab* introdujo el modelo XSeg para realizar esta segmentación de forma manual, en donde primero se deberán marcar las caras más representativas del conjunto de datos. Bastará con seleccionar cierta cantidad de muestras para realizar su entrenamiento. Cuando el entrenamiento del modelo XSeg es finalizado, se reemplazarán las máscaras predeterminadas por las máscaras generadas por el modelo XSeg para así poder continuar con la siguiente fase. Se deben seguir los siguientes 3 pasos en caso de querer utilizar el modelo XSeg para la realización de sus *deepfakes*.

XSeg - Enmascaramiento de los rostros destino

Se usará el archivo **5.XSeg) data_dst mask – edit.bat** para enmascarar algunas caras con diferentes ángulos y/o expresiones, o bien para eliminar alguna obstrucción en el video destino. Para un mejor resultado se deben marcar pequeños puntos del contorno rostro como se ve en la Figura 3.11.

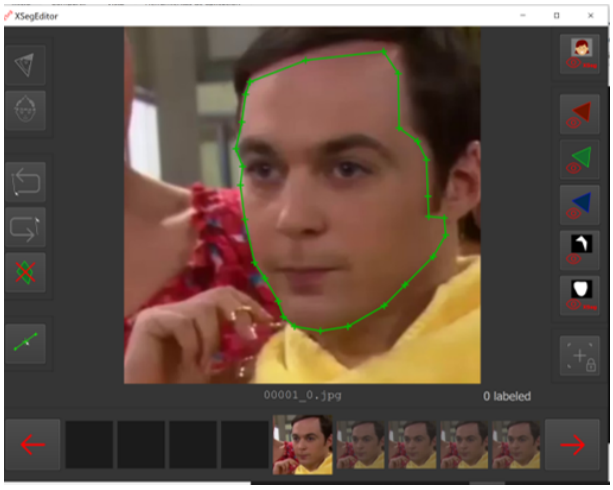


Figura 3.11: Enmascaramiento de video destino.

XSeg - Enmascaramiento de los rostros fuente

Se utilizará **5.XSeg) data_src mask – edit.bat** para enmarcar ahora los rostros del video fuente, un ejemplo de esta acción se muestra en la Figura 3.12.

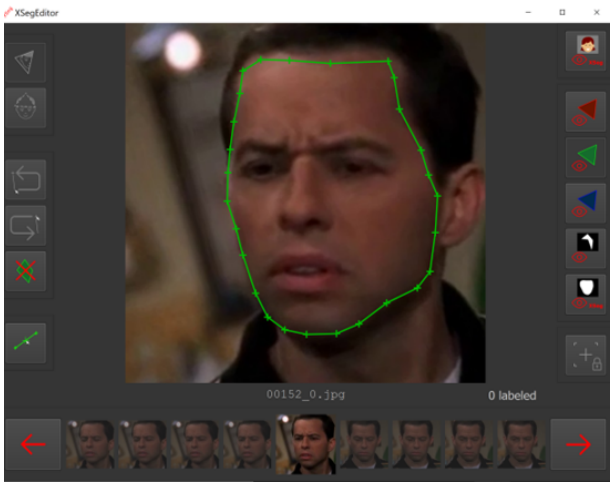
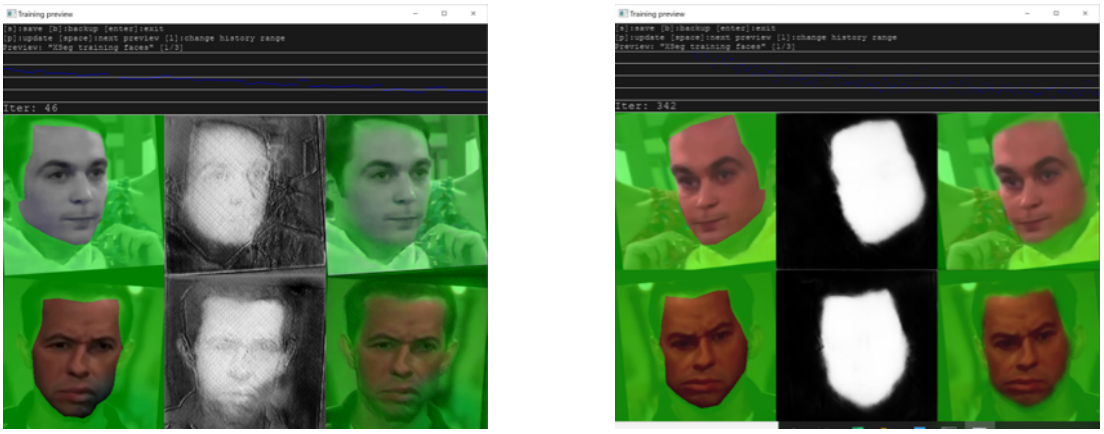


Figura 3.12: Enmascaramiento de video fuente.

XSeg- Entrenamiento del modelo XSeg

Una vez marcados los rostros se entrena el modelo XSeg con **5.XSeg) train.bat** y aparecerá una vista previa del proceso del entrenamiento como se muestra en las Figuras 3.13a

y 3.13b. Al terminar el entrenamiento, se aplicará la mascara entrenada **5.Xseg) trained mask for data_src-apply.bat** y **5.Xseg) trained mask for data_dst-apply.bat**.



(a) 46 epochs. (b) 342 epochs.

Figura 3.13: Entrenamiento modelo XSeg.

3.2.2. Entrenamiento

Esta segunda fase es la más importante para la generación de videos *deepfake*, para realizar el entrenamiento se cuenta con dos modelos, el modelo SAEHD recomendado para GPU con al menos 6GB de VRAM y Quick96 dedicado para GPU con 2 a 4 GB de VRAM [8].

Modelo Quick96

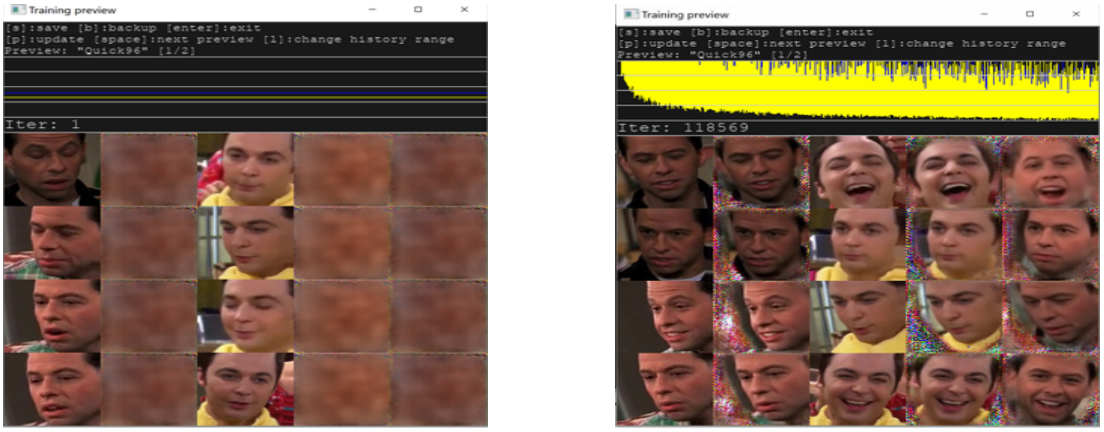
Este modelo se puede utilizar con tarjetas de gama baja, es un modelo no ajustable por lo que conlleva que tenga ciertos parámetros fijos como lo son que la tarjeta de video admita una resolución facial de 96 x 96, trabaja con caras completas, el tamaño del lote es de tamaño 4 y la arquitectura que utiliza es la arquitectura DF-UD. Al momento de realizar entrenamiento con este modelo la única opción a elegir es si entrenar con CPU o GPU o bien ambos.

Modelo SAEHD

En cambio, el modelo SAEHD es más potente y configurable, requiere una tarjeta de al menos 6GB de VRAM y admite resoluciones de hasta 512 x 512. Este modelo cuenta con más configuraciones (ver Anexo C para más información), es más tardado y genera menos iteraciones comparado con Quick96, sin embargo la resolución del rostro obtenido es mayor e iguala mejor la tonalidad del rostro, por mencionar algunas mejoras.

Entrenamiento con Quick96 o SAEHD

Se entrena el modelo con cualquiera de las dos opciones Quick96 **6)train Quick96.bat** o SAEHD **6)train SAEHD.bat**. Para este ejemplo se utilizó el modelo Quick96 y las Figuras 3.14a y 3.14b muestran la vista previa de como se desarrolla el entrenamiento por número de iteraciones, por ejemplo en la Figura 3.14a se ven los resultados en la primera iteración o época, en las columnas uno y tres se ven los rostros origen y destino, respectivamente. En las columnas dos y cuatro se ven sus reconstrucciones que hasta este punto parecen manchones. La Figura 3.14b muestra un avance del entrenamiento, en donde en las columnas dos y cuatro se pueden el avance de las reconstrucciones, las cuáles ya tienen un rostro mejor formado y finalmente en la última columna aparece el rostro aprendido, en esta se muestra el rostro fuente con las expresiones del rostro destino.



(a) 1 epoch. (b) 118569 epochs.

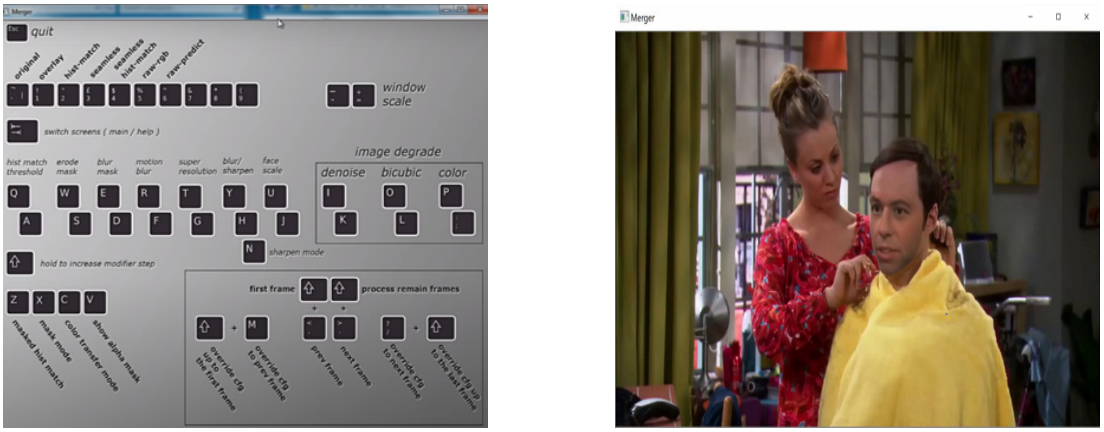
Figura 3.14: Entrenamiento Quick96/SAEHD.

3.2.3. Conversión

Fusión de rostros sobre video destino

Una vez que el modelo seleccionado finalice el entrenamiento se deberá realizar la fusión de las caras aprendidas sobre los fotogramas originales, para ello se utilizará el archivo **merge SAEHD.bat** que abrirá la fusión interactiva en la cual se podrán aplicar ajustes para hacer más realista el video *deepfake*. Se deberán realizar las modificaciones que se crean necesarias para que se vea creíble. Los ajustes más utilizados consisten en desenfocar la cara aprendida, escalarla de acuerdo a la cara destino, controlar el tamaño de la máscara, difuminar su borde para que este no sea tan notorio y utilizar un modo

de transferencia de color, el cual permite que el color de la cara aprendida coincida mejor con el color de piel de la cara destino. Después de aplicar todos los ajustes convenientes en este proceso de fusión, se obtendrá cada uno de los fotogramas del video destino, pero ahora con la nueva cara dentro de una carpeta llamada *Merged*. La Figura 3.15a muestra el menú de configuraciones del mezclador y la Figura 3.15b muestra la vista en donde se pueden apreciar los cambios realizados.



(a) Configuraciones.

(b) Ejemplo de modificación del *deepfake*.

Figura 3.15: Mezclador *DeepFaceLab*.

Conversión de fotogramas a video

Una vez guardados los cambios y para finalizar con esta fase, se deben convertir los cuadros a video nuevamente con **8)merged to mp4.bat**. El resultado final se encontrará dentro de la carpeta *workspace* como se muestra en la Figura 3.16.

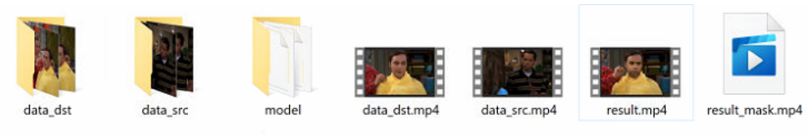


Figura 3.16: *deepfake* resultante en la carpeta *workspace*.

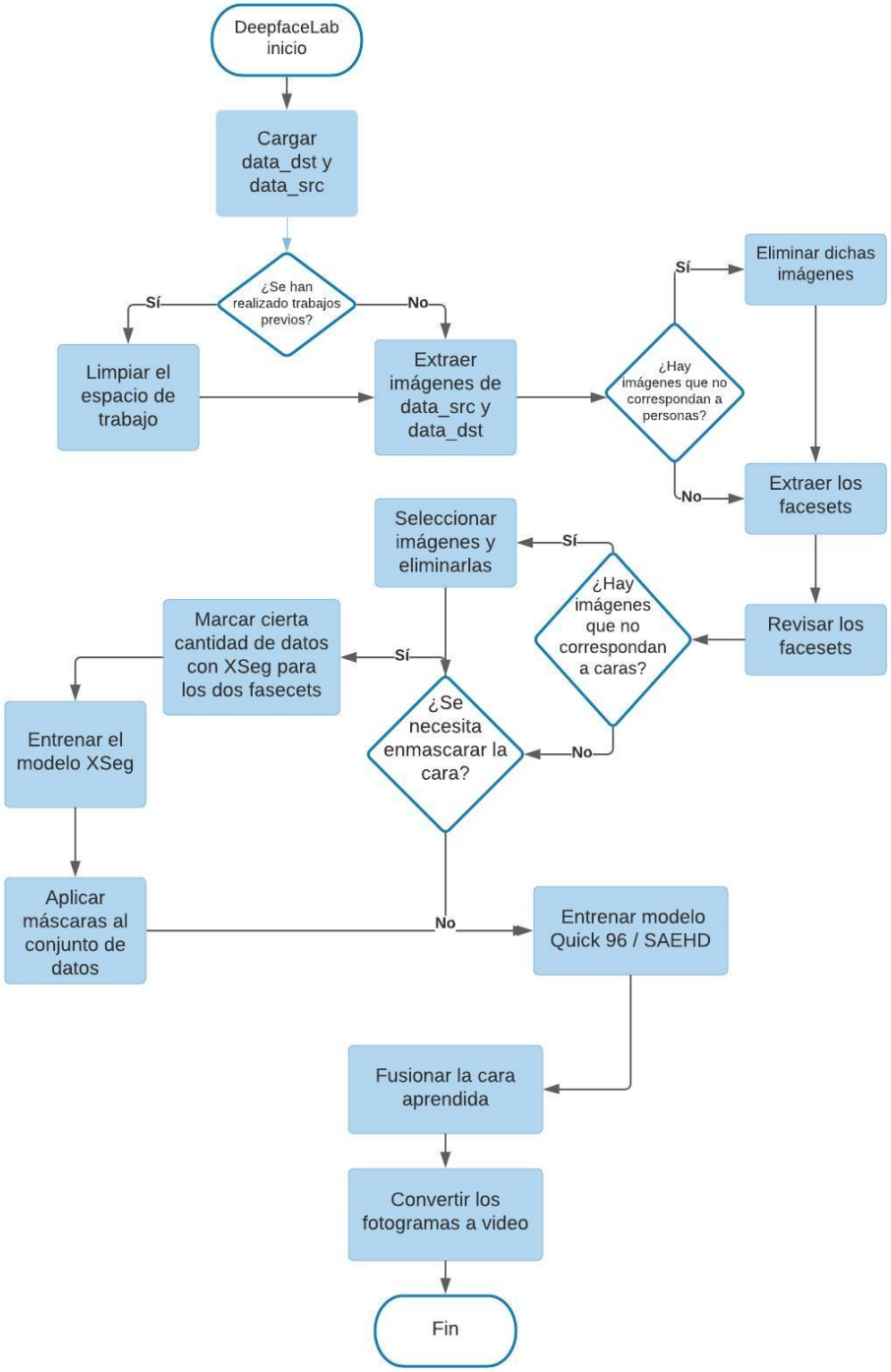


Figura 3.17: Diagrama de flujo de *DeepFaceLab*.

Capítulo 4

Experimentación y resultados

En esta sección se presentan los experimentos realizados utilizando varios conjuntos de videos con distintas características, para poder estimar los factores que influyen en la realización de *deepfakes*. En este apartado se mencionan los conjuntos de datos utilizados, las características consideradas, el trabajo hecho con cada una de ellas y algunos resultados con base en imágenes. Sin embargo, este trabajo de investigación se fundamenta en la creación de videos por lo que los resultados se pueden encontrar en los enlaces de cada prueba.

4.1. Conjunto de datos

Como se ha mencionado anteriormente, para poder generar *deepfakes*, es indispensable disponer de dos videos, un video fuente, del cual se tomará el rostro fuente y un video destino, en donde se colocará el rostro fuente con un cuerpo y fondo diferente, la Figura 4.1 muestra un ejemplo de este proceso.

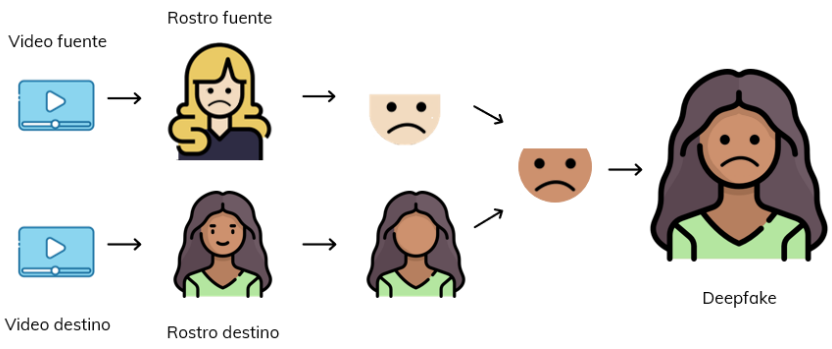


Figura 4.1: Videos de entrada para la generación de *deepfakes*.

En este trabajo se utilizaron videos de celebridades de la plataforma YouTube, algunos videos del conjunto de datos Celeb-DF-V2, y también se usaron videos domésticos.

4.2. Etapas de experimentación

4.2.1. Pruebas iniciales

Conjunto de datos

- Videos tomados de personas del espectáculo en entrevistas y discursos.

Características

- Videos con poco movimiento.

De acuerdo a la Figura 4.1 la prueba más sencilla que podría realizarse y la que permitiría una compatibilidad perfecta, es usar el mismo video como fuente y destino. Debido a que, de esta forma se puede ver el mejor resultado que puede entregar *DeepFaceLab*. Por esta razón, se realizó una etapa de prueba para familiarizarse con el sistema *DeepFaceLab* y obtener conocimientos sobre su uso y el de las herramientas con las que cuenta para la realización de *deepfakes*. En esta etapa se experimentó con el mismo video con el objetivo de observar qué tan bueno era el resultado en el caso más sencillo. La Figura 4.2 muestra el rostro de la periodista Brianna Keilar, el cual fue tomado como rostro fuente y rostro destino. Este video fue obtenido de Youtube, en el cual Brianna se encuentra presentando las noticias siempre viendo hacia en frente y no realiza gran movimiento del rostro. Este tipo de videos en donde se muestra a comentaristas o reporteros en primer plano dirigiéndose hacia la cámara se conocen como *talking heads*. La Figura 4.3 muestra el resultado de esta primer prueba[1], la cual cumplió con el objetivo de aprender a utilizar el sistema, además de que el *deepfake* fue considerado bueno. No se realizaron muchas modificaciones para obtenerlo, es decir, no se modificó el tono de piel, ni se erosionó o difuminó el rostro obtenido para que se viera mejor. De igual forma, la persona no realiza muchos movimientos y/o expresiones de modo que el resultado en relación a estos también fue considerado bueno, por otro lado, es importante mencionar que la resolución final si es más baja a la del video original.

[1] https://youtu.be/d8rS6jRfUiU

(a) Rostro fuente.

(b) Rostro destino.

Figura 4.2: Pruebas iniciales - mismo video como fuente y destino.

Figura 4.3: Pruebas iniciales - *deepfake* obtenido de un mismo video.

La segunda prueba se realizó con videos de diferentes personas los cuales tuvieran mayor movilidad. La Figura 4.4a muestra el rostro fuente (Justin Trudeau) y la Figura 4.4b muestra el rostro destino (Tom Holland). Realizando el *deepfake* se debería mostrar el rostro del político Justin en el cuerpo del actor Tom, como se puede ver en la Figura 4.5. Para obtener este producto se tuvo que cambiar el tono de piel para que el rostro fuente se acoplará con el del rostro destino, y se difuminó un poco. Se pudo lograr el intercambio de rostros como se muestra en el video de salida[2]. A comparación de los resultados obtenidos por la primera prueba, en este resultado se puede observar que la sincronización del audio

[2]https://youtu.be/XRh5jRn4cBg

con el movimiento de los labios no es buena, y se presentaron algunas manchas brillosas en el lado derecho del rostro y en la nariz.

(a) Rostro fuente.

(b) Rostro destino.

Figura 4.4: Pruebas iniciales - diferentes videos como fuente y destino.

Figura 4.5: Pruebas iniciales - *deepfake* obtenido de dos videos distintos.

Estos resultados cumplieron los objetivos iniciales de adaptarse al sistema y generar videos falsos, pero las manchas generadas en el resultado de esta última prueba generaron preguntas de la razón de este problema. De tal manera que, se analizaron los videos fuente y destino originales de la segunda prueba y se concluyó que uno de los factores influyentes en el resultado es la diferencia de la iluminación de éstos. Se hicieron más pruebas que generaron más incógnitas y se decidió estudiar las características que se creía eran las que tenían impacto sobre el resultado. A continuación se enlistan estas características.

Relativo a los videos de entrada

- Iluminación

- Tamaño del rostro / Diferente encuadre
- Afinidad de rostros
- Tono de piel
- Movimiento y rotación del rostro en distintos ángulos

Relativo a la red neuronal

- Tiempo de entrenamiento
- Hiperparámetros de la red neuronal

En cada una de las subsecciones siguientes se presentarán las etapas de experimentación realizadas con cada una de estas características.

4.3. Relativo a los videos de entrada

4.3.1. Iluminación

En las pruebas se observó que la iluminación del entorno de los videos originales influía en el resultado. Por lo que se decidió hacer un estudio sobre el tipo de iluminación que podrían tener los videos y la dirección de la fuente de iluminación así como la intensidad.

Se propuso la siguiente notación para describir la iluminación utilizada en los videos: posición de la iluminación horizontal, posición de la iluminación vertical, tipo de iluminación. Para seleccionar el tipo de iluminación, se encontró que existe una gran variedad de formas de hacerlo, sin embargo, aquí se mencionan los utilizados en los videos.

- **Directa**: Emite directamente desde una fuente de luz concentrada. Este tipo de luz se utiliza para resaltar elementos o personas en el medio donde se encuentren. El tipo de sombra que genera es densa y definida. Es similar a la iluminación directa del sol.

- **Difusa**: Crea una iluminación más homogénea que da claridad a todo el lugar, sin generar sombras. Es similar a la iluminación que viene de una ventana.

- **Indirecta**: La dirección en la que se encuentra este tipo de iluminación es hacia la parte de arriba, es decir hacia el techado del cuarto o habitación, con el objetivo de obtener una luz más tenue. Es similar a la luz en exteriores en un día nublado.

Ahora, para describir la posición de la iluminación se tomaron ciertos ángulos, empleando como referencia el modelo mostrado en la Figura 4.6. En ésta se identifica la posición de la fuente de luz dividiendo las circunferencias una, en eje horizontal y otra en eje vertical. Su funcionamiento es similar al de un reloj, las posiciones de forma horizontal van desde 1H a 12H, y de forma vertical van desde 1V a 12V.

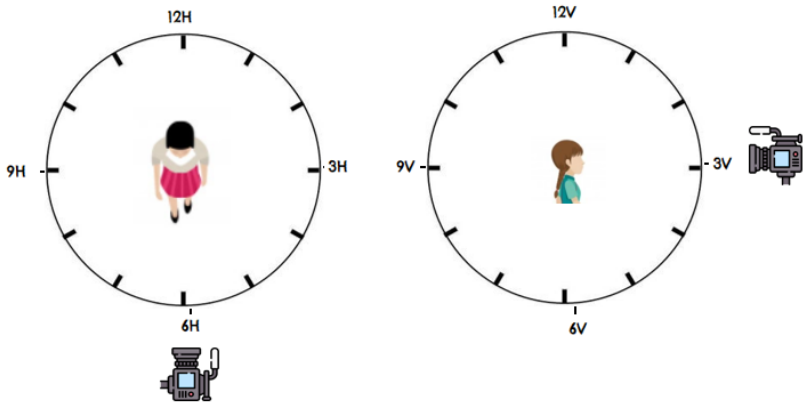


Figura 4.6: Ángulos de iluminación

De esta forma, para cada video se analiza la dirección de la fuente de luz y su tipo. Para el video fuente se usó la notación (X1, Y1, Z1) y para el video destino (X2, Y2, Z2). Posteriormente, se decidió calcular una distancia entre ellas con la fórmula 4.1.

$$D_E(P1, P2, P3) = \sqrt{(X2 - X1)^2 + (Y2 - Y1)^2 + (Z2 - Z1)^2} \tag{4.1}$$

Mientras más pequeño es el valor de la distancia, los rostros tienen una iluminación más parecida. Sin embargo, los tipos de iluminación no tenían un valor numérico, por lo que se propuso la siguiente escala para poder hacer cálculos:

- Directa - 2
- Difusa - 4
- Indirecta - 6

Es decir, si el video fuente presentaba una iluminación tipo: (6H, 8V, directa) y el video destino una iluminación (3H, 1V, difusa) para realizar los cálculos de la distancia se tendrían los valores (6, 8, 2) y (3, 1, 4) respectivamente. Ahora, la distancia entre estos videos se calcularía de la siguiente forma: las características del video fuente serían (X1, Y1, Z1) y las del video destino (X2, Y2, Z2), conforme a la fórmula 4.1 el cálculo sería el siguiente:

$$\begin{aligned} D_E(P1, P2, P3) &= \sqrt{(3-6)^2 + (1-8)^2 + (4-2)^2} \\ &= \sqrt{(-3)^2 + (-7)^2 + (-2)^2} \\ &= \sqrt{(9) + (49) + (4)} \\ &= \sqrt{(62)} \\ &= 7.87 \end{aligned} \tag{4.2}$$

De esta forma, se estimó qué tan compatibles eran las iluminaciones del video origen y destino, en las pruebas restantes que se describen a continuación.

Ensayo: Videos de celebridades en el mismo y diferentes set de grabación.

Se dedujo que si se seleccionaban videos de personas que se encontraran en un mismo programa, entrevista, set de grabación, entre otros, éstos tendrían la misma iluminación, por otro lado, si se escogían videos donde las personas estuvieran en sets distintos, la iluminación podía variar. La Figura 4.7 muestra el conjunto de datos en donde se tienen dos personas distintas situadas en el mismo show. Estos videos tienen la misma fuente de iluminación, con característica (8H, 3V, directa) y usando la fórmula 4.1 se obtiene que su distancia es 0. La Figura 4.7a muestra el rostro fuente (Emma Watson) y la Figura 4.7b muestra el rostro destino (Gal Gadot). Por otro lado, la Figura 4.8 muestra un fotograma del video *deepfake* resultante[3] donde se puede ver que el producto tiene un buen acoplamiento de los movimientos faciales a pesar de que los atributos de la persona fuente son más pequeños que la persona destino. Se tuvo que ajustar el tono de piel pero no fue muy complicado y a diferencia del video entre Justin Trudeau y Tom Holland ahora no se encuentran manchas brillosas, tampoco se ve la diferencia de tonalidades entre rostros, ni la línea de la máscara (comúnmente muy visible en la frente), sin embargo estos videos contenían diferentes movimientos y ángulos de la cabeza los cuales no pudieron ser aprendidos del todo, por lo que por un momento en el video el rostro se pierde.

[3]https://youtu.be/pAgfZdASntM

(a) Rostro fuente.

(b) Rostro destino.

Figura 4.7: Misma iluminación con rostros diferentes.

Figura 4.8: *deepfake* obtenido de dos rostros distintos con misma iluminación.

Por otro lado, se realizaron pruebas en donde las personas involucradas tuvieran iluminación distinta o bien, se encontraran en diferentes locaciones. Las Figuras 4.9 y 4.10 muestran el conjunto de datos utilizado para estas pruebas y la Tabla 4.1 muestra el tipo de iluminación estimado para los videos, fuente y destino, y la distancia calculada entre estos. Cabe mencionar que, las personas en el conjunto de datos 1 tienen tonalidad distinta y la tonalidad de las personas del conjunto 2 es más similar.

(a) Rostro fuente.

(b) Rostro destino.

Figura 4.9: Videos origen y destino con iluminación distinta, (distancia:8.3)

(a) Rostro fuente.

(b) Rostro destino.

Figura 4.10: Videos origen y destino con iluminaciones más compatibles, (distancia:3.0)

Tabla 4.1: Características de iluminación de los videos originales.

Figura	Video fuente	Video destino	Distancia
Figura 4.9	(4H, 2V, directa)	(12H, 1V, difusa)	8.30
Figura 4.10	(1H, 1V, difusa)	(3H, 2V, directa)	3.0

Las Figuras 4.11a y 4.11b muestran los resultados obtenidos. En el primer resultado[4], mostrado en la Figura 4.11a se ve un rostro distorsionado con iluminaciones en el contorno, aparte de la diferencia de tonalidades, el rostro fuente tiene mayor calidad y diferente textura que el video destino. Parece evidente que la luz del video fuente pega en el rostro, generando sombras en un costado. El segundo resultado[5], mostrado en la Figura 4.11b,

[4]https://youtu.be/vhAMV3Lljq4

[5]https://youtu.be/KjR9wv0g03A

se realizó con celebridades de tonalidad de piel muy parecida sin embargo, se presentan en diferentes locaciones, una en un set pequeño para una entrevista y la otro en un set de película, de modo que el resultado no se ve muy afectado pero si resaltan las partes principales de la máscara en la línea frontal y de menor forma en los laterales, por lo que el rostro aprendido (es decir, el rostro en el *deepfake*) se ve brilloso en algunas partes como la frente, las mejillas y la barbilla. En ambos resultados muestran muy marcada la línea de la máscara en la parte superior del rostro y/o tonalidades de piel las cuales no pudieron ser igualadas con la gama de colores que se ofrece.

(a) Distancia 8.3

(b) Distancia 3.0

Figura 4.11: *deepfakes* resultantes con dos tipos diferentes de iluminación.

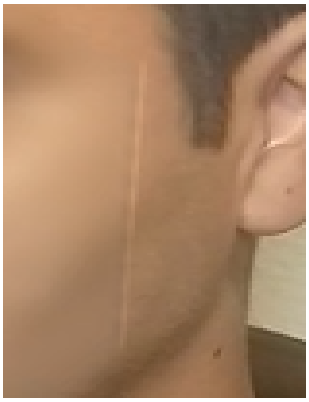
Ensayo: Videos domésticos con iluminación similar y poco movimiento del rostro.

Con el fin de disminuir las variaciones en iluminación se realizaron varios videos domésticos que incluían el rostro de algunas personas con fondo e iluminación similar y poco movimiento de la cara. Con estos videos se realizaron las siguientes pruebas:

- Pruebas sobre el mismo conjunto de datos
- Pruebas sobre diferentes conjunto de datos.
- Videos con 2, 4, 6 y 8 horas de entrenamiento.
- Modelo de entrenamiento Quick96.

Se grabaron videos con un fondo blanco, en donde la iluminación al rostro tratara de no generar sombras sobre el mismo con una iluminación estimada de (6H, 3V, directa) y haciendo un enfoque medio a la cara. Se optó por probar con el mismo conjunto de datos, es decir con un mismo video, esto para ver si los resultados con respecto a la iluminación y la realización de videos domésticos presentaba algún cambio. Por ende, la distancia de iluminación para esta prueba fue de 0. Los resultados obtenidos experimentando con el mismo video fueron buenos, como ya se había mencionado, la resolución del video si disminuye pero, los movimientos del rostro, tamaño y acoplamiento de los atributos se pueden considerar buenos, solo se presentó un caso en el que el individuo al momento de hacer una rotación hacía el lado derecho mostraba una línea blanca. La Figura 4.12a muestra el primer *deepfake*[6] en donde no se presentaron detalles en el rostro y la Figura 4.12b muestra el *deepfake*[7] donde si se pueden apreciar las líneas blancas en un extremos del rostro al girarlo.

(a) *deepfake* sin defectos.



(b) *deepfake* con defectos.

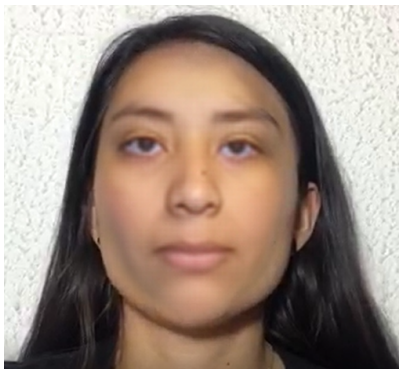
Figura 4.12: *deepfakes* domésticos con misma iluminación y mismos video origen y destino.

Se procedió a realizar experimentos con videos domésticos de diferentes voluntarios pero no se obtuvieron buenos resultados. En éstos, la cara fuente no embonaba con la cara destino, una se mostraba muy pequeña y la otra demasiado grande. Se sabe que por defecto esta herramienta determina el "mejor" tamaño, sin embargo, la proporción del rostro puede ser ajustada para que se acomode mejor. Lamentablemente en algunos casos este ajuste puede alterar los atributos del rostro y sus movimientos, además de mostrar una cara desproporcionada y menos natural. En estas pruebas al tratar de acomodar el

[6]https://youtu.be/DakHOCk3Gpg

[7]https://youtu.be/bqDpQNOGCPY

rostro se hicieron ajustes para que se viera real, no obstante, al hacer la cara pequeña se empezaba a ver el rostro destino, de tal forma que en la parte superior se veían dos cejas, una más arriba que otra o una más gruesa, de igual forma, cuando se realizaban movimientos con los labios estos no estaban acoplados. Se creyó que podía haber afectado al resultado que uno de estos videos fue grabado en formato horizontal y el otro vertical, aunque más adelante se volvió a hacer una prueba con videos grabados ambos en formato horizontal y aun así los rostros no embonaban bien. El video del autor contaba con un tipo de iluminación (6H, 3V, directa) y lo ideal era que ambos videos cumplieran con esta escala, sin embargo el otro video tiene un tipo de iluminación similar pero no idéntica con características (3H, 2V, directa) lo cual genera una distancia iluminación de 3.16. La Figura 4.13 muestra algunos de los resultados obtenidos en donde no se pudo igualar el color, es notable la diferencia de tonos de piel, el tamaño del rostro y la linea de la máscara en la frente. En la Figura 4.13a se muestra el *deepfake*[8] en el que se utilizó el video del autor como destino y en la Figura 4.13b presenta el *deepfake*[9] que se utilizó como fuente.



(a) Rostro fuente más pequeño

(b) Tono de piel distinto

Figura 4.13: *deepfakes* domésticos con iluminación similar y diferentes videos.

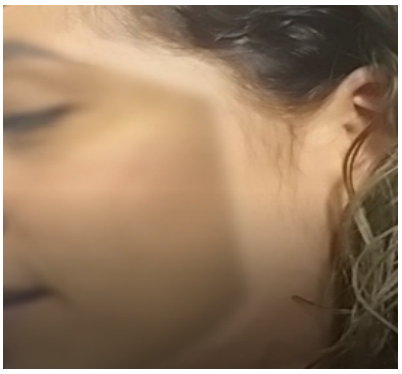
Ensayo: Videos domésticos con iluminación similar y movimiento amplio de los rostros

Continuando con la experimentación con videos domésticos, se grabaron videos que tuvieran una rotación mayor del rostro que los videos anteriores, esta vez la cabeza se giró aproximadamente 70° de derecha a izquierda, incluyendo también movimientos de arriba hacia abajo. Ambos videos fueron grabados de forma horizontal, y se optó por

[8]https://youtu.be/-_dVj7d6eUM
[9]https://youtu.be/IYt2N17rbYw

realizar un pre-procesamiento a los videos iluminándolos con el programa *Avidemux*, para determinar si pequeñas modificaciones podían ayudar a obtener mejores resultados aunque las iluminaciones fueran diferentes, como mayor difuminación de la línea de la máscara o bien que el color del rostro pudiera igualarse. De igual forma, se experimentó con el mismo video para observar el desempeño que tenían los resultados haciendo estas ediciones de iluminación. Los videos contaban con iluminación (6H, 3V, directa) y (3H, 2V, directa). La Figura 4.14a muestra el *deepfake* obtenido de la experimentación sobre el mismo conjunto de datos[10] donde se puede ver que el resultado es aceptable. Pero sobre diferentes conjuntos de datos los *deepfakes* no resultaron ser buenos, la Figura 4.14b muestra una parte del *deepfake*[11] donde se puede ver que se marcaba mucho más la diferencia del tono de piel por el video iluminado, así que se decidió entrenar estos videos sin iluminación sin embargo, el resultado tampoco fue favorable. Ambos videos presentaban una distancia de 3.16 igual que la anterior y quizá por la cercanía de la luz o por que al iluminar los videos no se hizo de una forma correcta, se obtuvieron estos rostros con más contrastantes en color.

(a) Mismo conjunto de datos.



(b) Diferente conjunto de datos.

Figura 4.14: *deepfakes* resultantes con iluminación modificada.

Ensayo: Videos de celebridades con iluminación y movimientos similares.

Se realizaron nuevas pruebas con videos de celebridades, para esta último ensayo se probó nuevamente con famosos que se encontraran en un mismo lugar de grabación, y se cuidó que tuvieran movimientos similares. La Figura 4.15 muestra el *deepfake* obtenido[12] de un video fuente y destino con misma iluminación (11H, 1V, directa), ambas

[10]https://youtu.be/3iUD8BmfIk0
[11]https://youtu.be/jt7FtkumBkA
[12]https://youtu.be/Pf9ZvqisWkI

personas fueron encontradas en el mismo canal Youtube de entrevistas. En cuanto a los resultados de iluminación muestra un rostro más natural, no se obtuvieron lineas laterales o superiores de otro color en el rostro, como anteriormente, ni divisiones de color muy contrastantes, ni parpadeos de diferentes tonalidades sobre el rostro.

Figura 4.15: *deepfake* de celebridades con misma iluminación y movimientos similares.

Los resultados obtenidos por la **iluminación** y rotación de rostros similares trajeron consigo ciertas dudas acerca de otros factores que podían ser influyentes en el resultado. Los ensayos 1 y la 3 mostraron que independientemente de la iluminación que se tenga, algunos ángulos y movimientos no pueden ser cubiertos en su totalidad y que el color de piel de los involucrados también puede afectar el *deepfake*. El ensayo 2 hizo suponer que la distancia a la que se encontraban las personas era motivo para que los rostros no embonarán o bien podría tratarse de que la afinidad de sus rostros era muy distinta, por lo que todos estos factores se estudiaron más a detalle y se presentarán en las siguientes secciones.

4.3.2. Modificaciones sobre el tipo de piel

El sistema de *DeepFaceLab* ofrece una gama de colores para modificar el tono de piel del rostro final de tal manera que pueda lograrse una combinación uniforme entre el cuerpo y la cara. Sin embargo, a pesar de contar con esta herramienta, los colores dados no siempre se asemejan al rostro destino y se puede notar la diferencia de tonalidades de piel en el resultado. En los *deepfakes* obtenidos con diferencias de **iluminación** se tuvo esta dificultad al intentar igualar los tonos de piel. Como se mostró en la sección anterior, en la Figura 4.16a se puede ver la combinación de dos tonos de piel distintos y la Figura 4.16b de tonos de piel semejante no obstante, en ambos se aprecia la diferencia de tonalidades en la parte frontal o en los ángulos laterales, o se muestra la máscara muy marcada.

(a) *deepfake* con diferente tono de piel.

(b) *deepfake* con tono de piel semejante.

Figura 4.16: Tono de piel en *deepfakes*.

Se creyó que el primer resultado se debía por la desigualdad de color, sin embargo el segundo tampoco fue muy convincente, por lo que decidió realizar algunos experimentos para ver qué tan influyente era el tono de piel y si al modificarlo se podían obtener mejores resultados.

Ensayo: Videos domésticos con tonos de piel similares.

En primer lugar, se realizaron pruebas con videos domésticos de familiares y conocidos que tuvieran una similitud en el tono de piel. Los resultados fueron aceptables, a diferencia de los obtenidos anteriormente, la Figura 4.17a muestra el primer resultado[13], se puede

[13]https://youtu.be/ddP6KWjHJ94

ver que el tono de piel es muy semejante lo cual hace no se vean diferencias de color en la parte frontal o en los laterales, además de que se logró una buena difuminación de la división de la máscara y ésta no es notoria en la parte frontal. La Figura 4.17b muestra el resultado del segundo experimento[14], donde se igualó lo más posible el tono de piel, sin embargo, se aprecia todavía una ligera disparidad en la parte frontal del rostro por encima de las cejas. Por lo que, se buscó una posible solución para disminuir aun más la diferencia de tonalidad y la línea de la máscara. La propuesta dada por usuarios que utilizan el sistema fue realizar el entrenamiento con el modelo SAEHD aproximadamente unas 500,000 epochs. Este modelo es muy tardado, sin embargo, se probó y se logró un mejor resultado. La línea de la máscara se ve más difuminada y hubo menos fallos en los laterales a diferencia del video anterior, cabe mencionar que la edición de ambos videos se realizó cuadro por cuadro para tratar de obtener el mejor resultado. La Figura 4.17c muestra el *deepfake*[15] final obtenido por este modelo.

(a) Modelo Quick 96 - primer video.

(b) Modelo Quick96 - segundo video.

(c) Modelo SAEHD - segundo video.

Figura 4.17: *deepfakes* domésticos con tonos de piel similar.

Ensayo: Videos de celebridades con tonos de piel similares.

En segundo lugar, se probó con dos actrices que tuvieran tono de piel semejante. La Figura 4.18a muestra el rostro fuente (Eva Longoria) y la Figura 4.18b el rostro destino (Victoria Justice). El resultado obtenido fue bueno, ya que no se notaba ningún tipo de manipulación sobre el video, inclusive la iluminación que presenta en el rostro parece ser parte del maquillaje o de la iluminación del entorno sobre el rostro. La Figura 4.19

[14]https://youtu.be/dUbwTxtIg4s
[15]https://youtu.be/dMw_MF6BdDY

muestra este resultado[16].

(a) Rostro fuente.

(b) Rostro destino.

Figura 4.18: Conjunto de datos de celebridades con tono de piel similar.

Figura 4.19: *deepfake* de celebridades con tono de piel semejante.

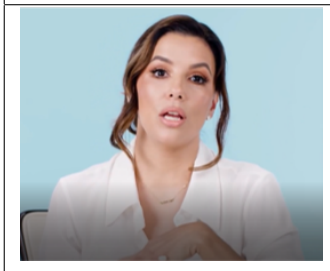
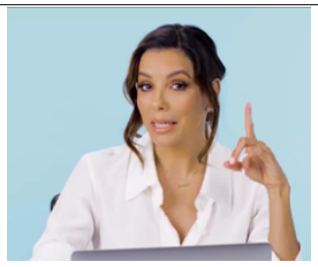
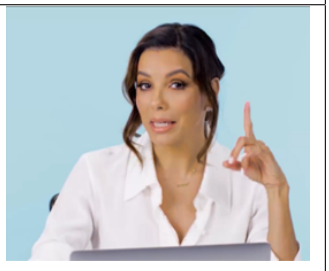
Como este resultado fue favorecedor, se tomó el conjunto de datos para realizar modificaciones sobre el tono de piel y observar su comportamiento. Para ello se intentó cambiar el tono de piel con un algoritmo basado en la detección de puntos faciales, sin embargo, los resultados obtenidos por éstos arrojaban tonos de piel no deseados o muy amarillentos, encima de que se debía cambiar el tono de piel a todas las imágenes del conjunto de datos, lo cual lo hacia un proceso tardado.

Por lo que se buscó otra opción que permitiera hacer la modificación de forma más rápida, para ello se utilizó el editor de video *After Effects* el cual permitió realizar las

[16]https://youtu.be/0A_8bAGzKx8

modificaciones, pero al igual que el primer método de igualación de tonos de piel fue un proceso complicado. Como primera modificación a la piel de la actriz Victoria Justice se hizo un poco más clara y a la piel de Eva Longoria más morena. Como segunda modificación se utilizó el mismo color de Eva para poder obscurecer más a Victoria, aunque a pesar de que utilizara el mismo color en ambas el tono de piel "no pigmentó" de la misma manera sus rostros, ya que uno se nota un poco más anaranjado. La Tabla 4.2 muestra el color en RGB de cada una de las modificaciones realizadas.

Tabla 4.2: Ajustes de tono de piel.

Original	Modificación 1	Modificación 2
		
	# F0D3C3	# e68b58
		
	# e68b58	# e68b58

Se puede decir que realizar modificaciones sobre el tono de piel no produce resultados significativos. La Figura 4.20a muestra el resultado de la primera modificación[17] en donde una de ellas era más morena que otra y la Figura 4.20b el *deepfake*[18] obtenido donde ambas son morenas. Ninguno de los resultados muestra líneas en la parte frontal o lateral muy marcadas, la unión de ambos rostros a pesar de su diferencia de color fue buena. Solamente es importante mencionar que en el resultado de la primera modificación en el *deepfake*, se alcanzan a apreciar ciertos parpadeos que en el video original no se presentaban, parece que estos fueron generados por que alguna imagen del conjunto de datos fuente no se editó correctamente.

[17]https://youtu.be/HLkYLU2Z-jk
[18]https://youtu.be/WxlUDGSGsYs

(a) *deepfake* con aclarado.

(b) *deepfake* con obscurecimiento.

Figura 4.20: *deepfakes* obtenidos con modificaciones del video original.

Ensayo: Videos de celebridades con tonos de piel diferente.

Finalmente, se realizó una prueba con dos personajes de tonalidad distinta, en la Figura 4.21 se muestra el conjunto de datos utilizado. Esta vez no se hicieron modificaciones sobre el tono de piel, estos videos fueron grabados por el mismo programa de entrevistas, por lo cual se asumió la iluminación es la misma. Se entrenó con los dos modelos para observar la diferencia ya que SAEHD logró semejar más el tono de piel en las pruebas anteriores.

(a) Rostro fuente.

(b) Rostro destino.

Figura 4.21: Conjunto de datos de celebridades con tonalidad de piel distinta.

La Figura 4.22a muestra el resultado del entrenamiento con el modelo Quick96[19] y la Figura 4.22b el resultado con el modelo SAEHD[20]. En el primero se alcanza a apreciar

[19]https://youtu.be/tMh8g6LpB-0

[20]https://youtu.be/1vdirmGWbFI

una luminosidad y una diferencia de color en la parte de la frente y en los lados del rostro sin embargo, el modelo SAEHD si asemeja más el tono de piel. En este experimento se tuvieron que realizar difuminaciones para hacer el rostro más uniforme, pero al realizar esta acción la cara se hace más pequeña de los alrededores, de tal forma que en este video hizo que se notara un poco más la barba del personaje destino en la parte inferior izquierda.

(a) *deepfake* obtenido por Modelo Quick96. (b) *deepfake* obtenido por Modelo SAEHD.

Figura 4.22: *deepfake* de celebridades con tono de piel distinto y misma locación.

Con base en estos experimentos se estableció que no es necesario realizar modificaciones a menos que se usen dos personas con tonos de piel muy diferentes, o cuando *DeepFaceLab* no pueda igualar estos tonos. Si se quiere igualar el tono de piel se debe tener en cuenta que se debe tener cierta habilidad en el manejo de softwares de diseño y que este preprocesamiento es tardado. Lo que se recomienda o es deseable es que los conjuntos de datos que se utilicen sean semejantes en tono de piel o revisar el resultado en un tiempo corto de ejecución, por ejemplo unos 75,000 epochs para ver si haciendo modificaciones con la gama de colores que ofrece *DeepFaceLab* puede lograr igualarse.

4.3.3. Tamaño del rostro / Diferente encuadre

Al realizar *deepfakes* con videos domésticos, se obtuvieron rostros resultantes que no eran proporcionales al rostro destino. En uno el rostro destino sobresalía, y al haber sido ajustado con la herramienta de *DeepFaceLab* los atributos de la cara por ejemplo, cejas, boca y ojos no se acoplaban correctamente, lo cual desacreditaba el *deepfake*. Se infirió que un factor influyente pudo ser el tamaño de los rostros originales, así que se realizó una segunda prueba con un video grabado a menor distancia. Sin embargo, en este resultado se obtuvo un rostro más grande. Por lo que se decidió experimentar con videos en donde el tamaño del rostro destino y fuente fueran diferentes, es decir que, tuvieran diferente encuadre.

El encuadre de video determina lo que entra dentro de la grabación y para definirlos se utilizan los tamaños de planos. En [11] mencionan ocho planos, pero a continuación se mencionan los utilizados en estas pruebas.

- *Close-up (CU)* - Primer plano.
- *Medium Close-up (MCU)* - Plano medio corto.
- *Medium Shot (MS)* - Plano medio.

Close up shots o primeros planos. Muestran principalmente ojos, boca y un encuadre de la persona encima de la clavícula. Los cineastas utilizan los primeros planos para enfatizar las emociones y despertar la atención de la audiencia. *Medium shots* o tomas medias. Generalmente se enfocan en el parte superior del cuerpo de los personajes, pueden ocupar la pantalla con un tercio de su cuerpo y permite tanto seguir las acciones en las que participan los personajes y ver sus reacciones y emociones. Para el plano *medium close- up* se enfoca de medio torso hacia arriba y *medium shot* de la parte de la cadera hasta la cabeza.

La Figura 4.23 muestra los tipos de encuadre.

(a) *Close Up* (b) *Medium Close Up* (c) *Medium Shot*

Figura 4.23: Tipos de encuadre

Para esta característica solo se realizó una etapa de prueba con las siguientes combinaciones de encuadre (video fuente y destino).

***Medium shot - Close up*:** Realizar este tipo de combinaciones son las menos recomendables debido a que en primer plano se puede apreciar fácilmente todo tipo de detalles como líneas, fallos en los movimientos de los labios, ojos, etc. Además de que en los resultados obtenidos los rostros no se ven naturales, la textura de la piel se ve borrosa. Además sus atributos como ojos y barba no son claros y se notan un tanto difuminados. En la Figura 4.24 se puede ver el conjunto utilizado, el rostro fuente en 4.24a con un encuadre *medium shot* y el rostro destino en 4.24b con un *close up*. Por otro lado, la Figura 4.25 muestra el resultado[21] del *deepfake* final.

(a) Rostro fuente - *medium shot*. (b) Rostro destino - *close up*.

Figura 4.24: Conjunto de datos con encuadre *medium shot* y *close up*.

[21] https://youtu.be/AWEJhcvFfVU

Figura 4.25: *deepfake medium shot - close up.*

Close up – Medium shot: A diferencia de los primeros planos, los planos medios pueden servir muy bien para realizar *deepfakes*. En estos se puede ver una cara aprendida favorecedora, no contiene partes borrosas. No obstante, la cara puede ser un poco más grande de la cara destino (no en todos los casos), lo cual puede resolverse con un ajuste simple, difuminando los extremos del rostro fuente aprendido para que pueda combinarse con el rostro destino, o bien ajustando el tamaño del rostro sin olvidar revisar el resultado después de ajustarlo y realizar modificaciones por que cuando se realizan este tipo de cambios puede deformarse el resultado. La Figura 4.26 muestra el conjunto de datos utilizado y el resultado final[22] es mostrado en la Figura 4.27, en el cual se puede ver los atributos del rostro con mejor forma que el de primer plano y una piel más lisa.

[22]https://youtu.be/_k1t2wjiPUU

(a) Rostro fuente - *close up.*

(b) Rostro destino - *medium shot.*

Figura 4.26: Conjunto de datos con encuadre *close up* y *medium shot.*

Figura 4.27: *deepfake close up - medium shot.*

Medium shot – Medium close up: El resultado es bueno y la textura de la piel se muestra uniforme. Sin embargo, se debe de ser cuidadoso ya que el rostro fuente aprendido puede mostrarse más pequeño en algunos ángulos, por lo que se debe difuminar correctamente con la herramienta que ofrece *DeepFaceLab*, de tal manera que logre desvanecer la diferencia entre estos dos rostros, o de igual forma tratar de calibrar el tamaño del rostro sin que el resultado en movimientos sufra alteraciones. La Figura 4.28 muestra el conjunto de datos y la Figura 4.29 el *deepfake* obtenido[23] donde se muestra un ejemplo

[23]https://youtu.be/IxePzcS6qDs

en el que del lado derecho el rostro aprendido no cubre en su totalidad el rostro destino, por otro lado los demás atributos se ven proporcionados al tamaño del rostro y en una posición adecuada. Finalmente, como el tamaño del rostro es más pequeño, su textura no se ve difuminada.

(a) Rostro fuente - *medium shot.*

(b) Rostro destino - *medium close up.*

Figura 4.28: Conjunto de datos con encuadre *medium shot* y *medium close up.*

Figura 4.29: *deepfake medium shot - medium close up.*

Medium close up - Medium close up: Como se mencionó anteriormente, los resultados en los planos medios muestran mejores resultados. Este es un plano en el que se tiene un acercamiento al rostro. Sin embargo, se puede lograr un buen producto final sin

que los rasgos o la piel del rostro se vean desvanecidos. Esta combinación, fue la primera en realizarse es por ello que se seleccionó el mismo conjunto de datos como fuente y destino, para observar los cambios que podían tenerse al combinar encuadres. A pesar de ser el mismo video, se tomaron fragmentos diferentes para realizar el entrenamiento. La Figura 4.30 presenta el conjunto de datos y la Figura 4.31 ilustra la combinación obtenida de este *deepfake*[24] donde se puede ver un rostro natural.

(a) Rostro fuente - *medium close up.*

(b) Rostro destino - *medium close up.*

Figura 4.30: Conjunto de datos, ambos con encuadre *medium close u.*

Figura 4.31: *deepfake medium close up - medium close up.*

Haber realizado estos experimentos mostró que el encuadre puede generar rostros un poco más grandes o más pequeños que el rostro destino, sin embargo *DeepFaceLab* en su herramienta de fusión interactiva pondrá el tamaño de rostro que considere más

[24]https://youtu.be/M2rlltdWUdo

adecuado. Si se cree que este tamaño no es el mejor, el usuario podrá ajustarlo, pero se deberá asegurar que esta modificación no genere problemas en los demás atributos del rostro y se obtenga un buen *deepfake*. Por otra parte, se observó que los mejores encuadres con los que se puede trabajar son los planos medios.

4.3.4. Afinidad de rostros

Otra de las inquietudes que surgieron a partir de la etapa de prueba fue que el rostro fuente no era proporcional al rostro destino. Incluso cuando se hicieron modificaciones para tratar de ajustarlo, los atributos no siempre se acomodaron completamente bien. Para resolver esta incógnita, en primer lugar se intentó con la distancia a la que se encontraba el individuo, como se mostró en la subsección anterior **tamaño del rostro**, pero en estas pruebas los rostros fuente mostraron acoplarse al tamaño del rostro destino, además de que la resolución del rostro baja más cuando se hacen *deepfakes* a primeros planos, independientemente del encuadre que se escoja. Es por ello que se optó por probar con la similitud que tenía el conjunto de datos para observar el comportamiento de los resultados.

Ensayo: Videos domésticos de personas con parentesco consanguíneo.

Se decidió trabajar inicialmente con videos domésticos de familiares, ya que se supuso tendrían una similitud debido al parentesco. Se grabaron videos de demostración para que fueran replicados por familiares y conocidos con algún parecido. De estos videos se extrajeron las imágenes del rostro fuente y el rostro destino, para obtener el porcentaje de similitud entre los dos usando una aplicación llamada *Golden Ratio Face App*[25]; sin embargo el resultado daba un porcentaje general de similitud de todo el rostro y se analizó que lo que se buscaba era ser un poco más específico sobre sus atributos para saber si embonarían bien en otro.

Ensayo: Selección de videos de celebridades con sejemanza.

No obstante, trabajar con rostros de las personas involucradas en el proyecto y familiares implicaba grabar y enviar videos lo cual era una tarea compleja, además de que no se podía disponer del tiempo de los demás, de modo que, se decidió trabajar con personas del espectáculo ya que era mucho más fácil obtener una gran cantidad de videos de ellos. Para encontrar personas del medio artístico con algún parecido se usó la aplicación *Star by Face*[26], la cual permitía ingresar la imagen del rostro de alguien y arrojaba una pequeña lista de celebridades de ambos géneros con el nombre, fotografía y una barra que indicaba el porcentaje de semejanza como se muestra en la Figura 4.32.

[25]https://play.google.com/store/apps/details?id=com.golden.ratio.face&hl=en_US&gl=US
[26]https://apps.apple.com/us/app/star-by-face-celebs-look-alike/id1499633307

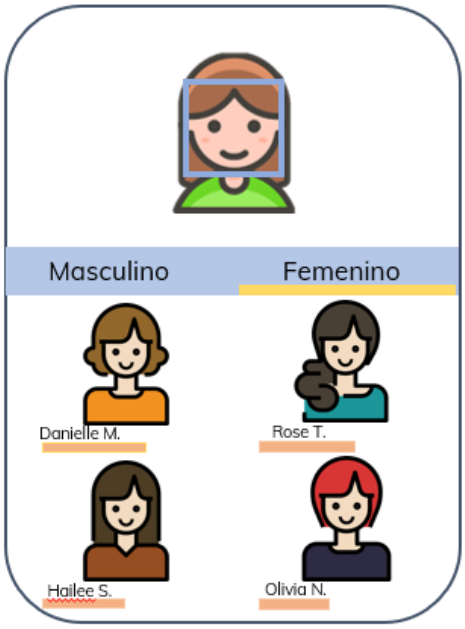


Figura 4.32: Aplicación *Star by Face*.

Esta aplicación fue muy útil y con ella se pudo realizar una primera prueba exitosa considerando la afinidad de rostros. Esta prueba fue la de la actriz Victoria Justice. La Figura 4.33 muestra algunas de las actrices con las que se encontró similitud, de la cual se seleccionó a la actriz Eva Longoria. El resultado del *deepfake* se muestra en la Figura 4.34, presenta un buen acoplamiento del rostro destino sobre el fuente, en el video se ve como se acomodan los atributos en su posición correcta, además de que el rostro se ve más natural y se pudo realizar una mejor difuminación para lograr un buen resultado. Pero de igual forma que el ensayo anterior se buscaba más precisión sobre los atributos.

Figura 4.33: Celebridades con similitud.

Figura 4.34: *deepfake* de celebridades con similitud.

Ensayo: Antropometría facial.

Debido a lo anterior, se consideró que era necesario desarrollar una herramienta adecuada a las necesidades del proyecto y se generó un algoritmo capaz de identificar la similitud de los atributos del rostro. Este algoritmo recibe las imágenes de dos rostros e identifica 67 coordenadas de las marcas faciales de forma gráfica y numérica. La Figura 4.35 muestra un ejemplo del gráfico obtenido al ejecutarlo, donde las marcas faciales del rostro 1 se grafican de color azul y las del rostro 2 de color rojo. La Tabla 4.3 muestra un ejemplo de las coordenadas obtenidas.

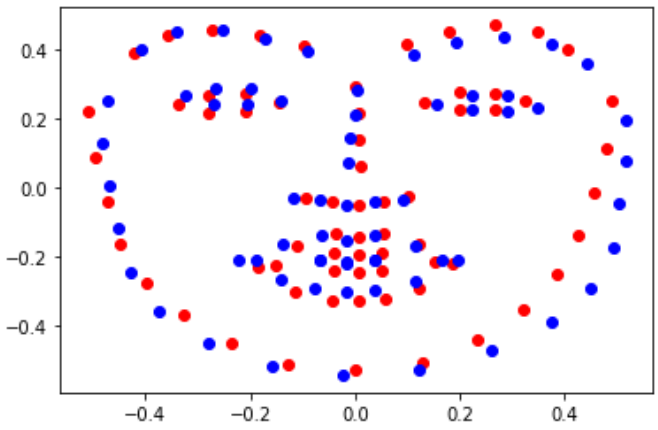
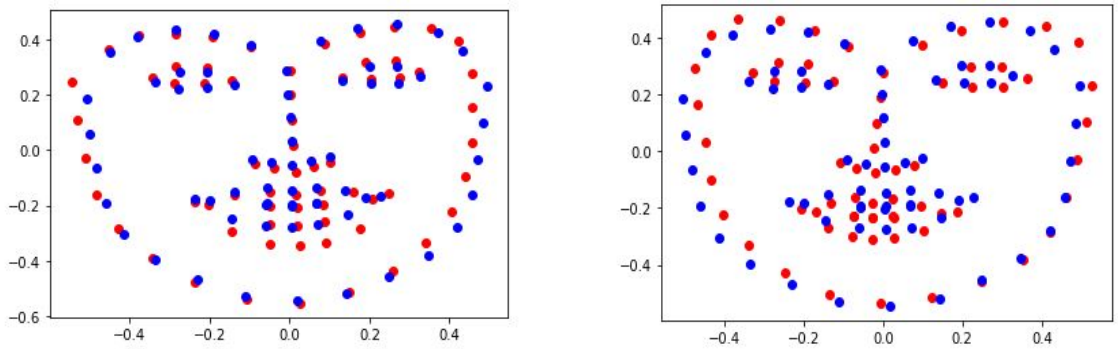


Figura 4.35: Marcas faciales obtenidas de dos rostros.

Tabla 4.3: Coordenadas de los 67 puntos obtenidos mostrados en la Figura 4.35

	X_1	Y_1	X_2	Y_2
0	-0.473700268	0.251945525	-0.392102397	0.326213375
1	-0.481999024	0.131322957	-0.421732026	0.204540371
2	-0.469550891	0.006809339	-0.432843137	0.079065086
3	-0.452953381	-0.11770428	-0.42543573	-0.046410199
4	-0.428057115	-0.242217899	-0.403213508	-0.168083203
5	-0.374115206	-0.358949416	-0.358769063	-0.282151644
...	...	...	...	...
67	-0.067061264	-0.211089494	-0.110620915	-0.183292328

Para seleccionar el conjunto de datos que tuviera mayor semejanza en las siguientes pruebas, se compararon los gráficos de sus marcas faciales. En principio estos esquemas se cotejaron de forma visual. La Figura 4.36 muestra un ejemplo donde se pueden ver las coordenadas faciales de dos conjuntos de datos, de éstos se escogería el conjunto de Victoria Justice y Eva Longoria, en vez del de Victoria y Mila Kunis. Sin embargo, ésto era un proceso subjetivo por lo que se recurrió a buscar una forma más precisa.



(a) Victoria - Eva Longoria.

(b) Victoria - Mila Kunis.

Figura 4.36: Comparación de las coordenadas faciales obtenidas.

La propuesta implementada usa los datos numéricos de la Tabla 4.3 para calcular la distancia Euclidiana entre dos puntos (P1,P2) de las coordenadas obtenidas, después se realiza la suma de todos estos valores y finalmente se calcula el promedio para hacer la comparación de manera numérica. La Tabla 4.4 muestra un ejemplo de los datos obtenidos.

De esta forma, mientras menor sea el valor obtenido, más cerca están las coordenadas de ambos rostros entre sí y es más fácil lograr un buen *deepfake*.

Tabla 4.4: Calculo de distancias entre dos rostros.

	X_1	Y_1	X_2	Y_2	Distancia Euclidiana
0	-0.473700268	0.251945525	-0.392102397	0.326213375	0.110335517
1	-0.481999024	0.131322957	-0.421732026	0.204540371	0.094830906
2	-0.469550891	0.006809339	-0.432843137	0.079065086	0.081045371
3	-0.452953381	-0.11770428	-0.42543573	-0.046410199	0.076420332
4	-0.428057115	-0.242217899	-0.403213508	-0.168083203	0.078186687
5	-0.374115206	-0.358949416	-0.358769063	-0.282151644	0.078316039
...	...	...	...	...	...
67	-0.067061264	-0.211089494	-0.110620915	-0.183292328	0.051673258
Total	-	-	-	-	3.867194525
Promedio	-	-	-	-	0.057719321

Ensayo: Cálculo de parecido con una pequeña base de datos.

El procedimiento que se realizó en la subsección pasada era largo, ya que se tenía que seleccionar un artista, buscar el parecido con otros y hacer las comparaciones uno a uno, por lo que se creó una base de datos pequeña, haciendo una búsqueda de personas del espectáculo de los cuales se pudiera obtener videos de forma más simple en la web, este trabajo se completó seleccionando celebridades del conjunto de datos Celeb-DF-V2. Una vez obtenidos los nombres de todas las personas involucradas, se procedió a descargar imágenes en donde estuvieran de frente y se realizó una modificación al algoritmo anterior para que solamente calculara las marcas faciales de un rostro.

Ahora las marcas resultantes se verían como en la Figura 4.37 y la Tabla 4.5.

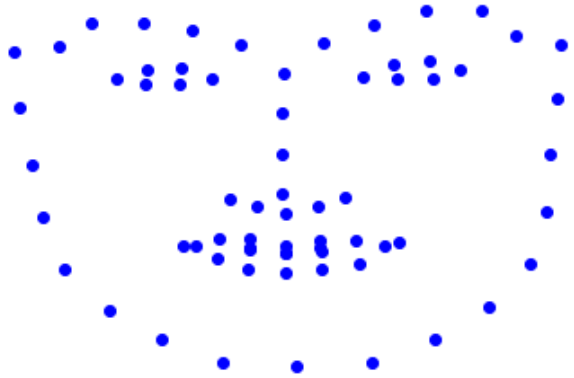


Figura 4.37: Marcas faciales obtenidas de un solo rostro.

Tabla 4.5: Coordenadas obtenidas de un solo rostro

	X_1	Y_1
0	-0.502406417	0.339314104
1	-0.491497326	0.18487395
2	-0.469679144	0.022711787
3	-0.447860963	-0.127867363
4	-0.407860963	-0.270724506
5	-0.327860963	-0.386554622
...	...	...
67	-0.069679144	-0.212809448

Así se obtuvieron las coordenadas de manera individual de aproximadamente cien famosos y se creó un algoritmo que pudiera obtener de manera rápida la diferencia numérica de estas coordenadas como se había realizado anteriormente, es decir, calculando la diferencia entre ellas con la distancia euclidiana, sumando el total y haciendo el promedio de estos datos. Finalmente, se creó una matriz de distancias, en donde se localizan los rostros que se asemejan más representados por valores pequeños, y los que menos se asemejan con valores grandes, además de que la obtención de estos datos sirvió para futuras pruebas, para terminar con esta etapa, se normalizaron los datos de la matriz de distancias con la ecuación 4.3.

$$Porcentajedesimilitud = 100 * (1 - \frac{x - xmin}{xmax - xmin}) \tag{4.3}$$

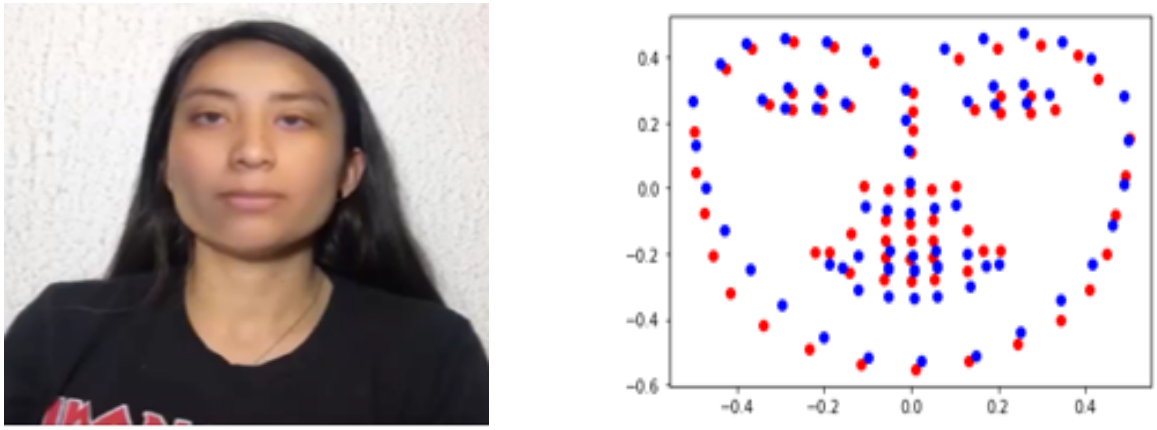
La Tabla 4.6 muestra un ejemplo de las distancias normalizadas. Se debe destacar que ahora los valores más cercanos a 100 representan a rostros más similares.

Tabla 4.6: Matriz normalizada.

Celebridad	Edward Norton	Emilia Clarke	Emma Stone	Ethan Hawke
Alfonso Cuarón	55.88	52.9	60.48	45.51
Alicia Vikander	49.06	58	72.99	49.9
Angelina Jolie	35.71	57.56	44.49	51.15
Anne Hathaway	46.52	60.38	41.21	47.78
Anni-Frid	30.25	38.48	60	33.88

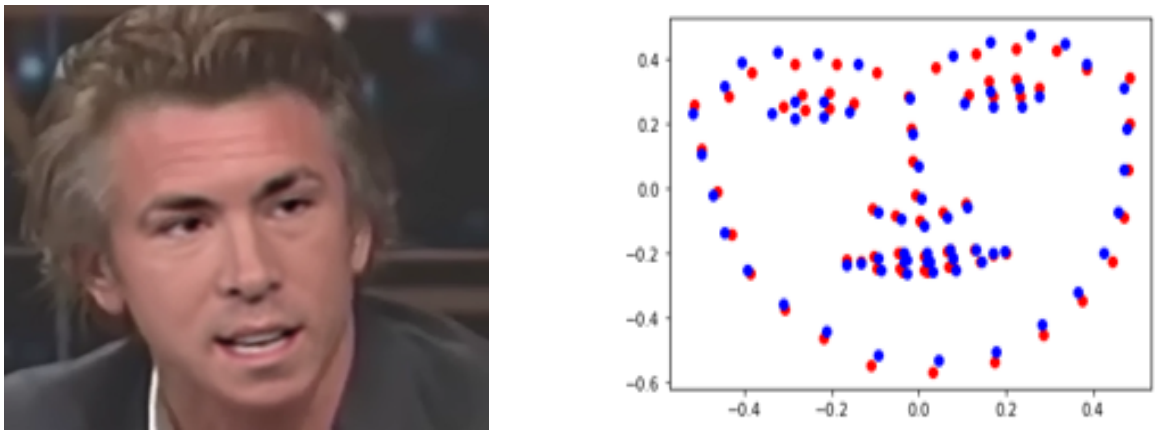
A continuación se muestran algunos resultados extremos obtenidos tomando en cuenta la afinidad. La Figura 4.38 presenta un *deepfake* obtenido de un conjunto de datos con marcas faciales las cuales no son semejantes. La Tabla 4.7 muestra que la similitud entre las personas involucradas dentro de este video fue de 5.33 %. En el *deepfake* obtenido[27] se puede ver un rostro que no embona correctamente con el rostro destino, a pesar de ser difuminado para tratar de eliminar las líneas de los alrededores, éste da la impresión de ser más pequeño y se puede observar en las sombras que genera a los lados, además de que cuando el rostro realiza ciertos movimientos, se puede ver el rostro destino desfasado o incompleto de lado. Por otro lado, se tiene la Figura 4.39 que muestra el resultado obtenido de un conjunto de datos con un 64.30 % de similitud[28]. En éste se puede observar un mejor acoplamiento de los atributos del rostro fuente en el destino, inclusive al hablar o mover la cabeza.

[27]https://youtu.be/-_dVj7d6eUM
[28]https://youtu.be/uwSuzDN4CoY



(a) (b)

Figura 4.38: *deepfake* obtenido de un conjunto de datos con marcas faciales no semejantes.



(a) (b)

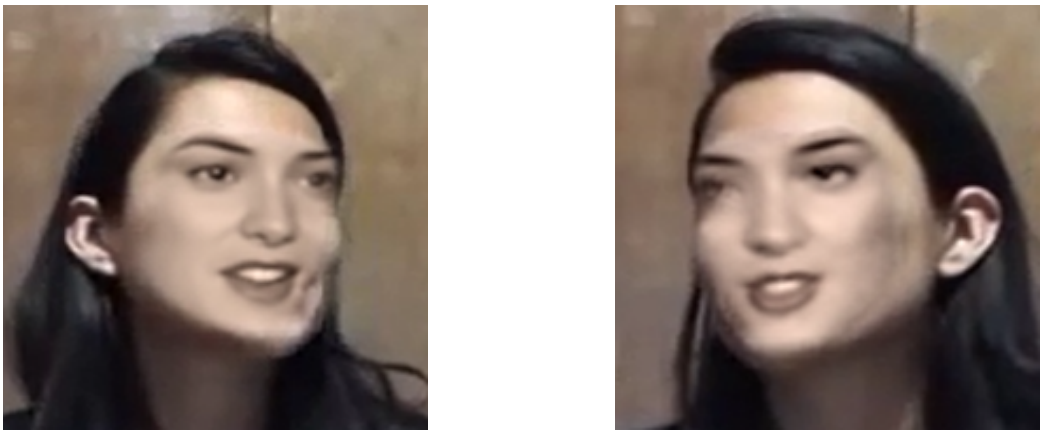
Figura 4.39: *deepfake* obtenido de un conjunto de datos con marcas faciales semejantes.

Tabla 4.7: Porcentaje de similitud

Conjunto de datos	% de similitud
Figura 4.38	5.33
Figura 4.39	64.3

4.3.5. Movimiento y rotación del rostro en distintos ángulos

Una de las recomendaciones que se encontraron en la web para la realización de *deepfakes*, era que el conjunto de datos que se utilizara tuviera movimientos semejantes para que en el entrenamiento el sistema pudiera aprender la cara en diferentes posiciones. Esto también se pudo ver en el primer ensayo de la característica de **iluminación**. Por lo que se probó más a fondo realizando un video moviendo el rostro hacia los lados y haciendo rotaciones del rostro hasta unos 50° aproximadamente, para comprobar esta teoría y ver si los movimientos podrían deformar el rostro. En efecto, los movimientos que se realizaron trajeron consigo caras desproporcionales y distorsionadas. Se asumió que este resultado se obtuvo debido a que el rostro del video fuente no realizaba ningún movimiento o rotación, si no que se encontraba viendo fijamente al entrevistador. La Figura 4.40 muestra ejemplos de los rostros del *deepfake* obtenido.



(a) Ejemplo 1. (b) Ejemplo 2.

Figura 4.40: Video prueba - rostro con movimientos.

Ensayo: Videos de celebridades con diferente movilidad del rostro.

Para la primera etapa de experimentación se utilizó el siguiente conjunto de datos:

- Videos en donde el rostro tenga poca movilidad.
- Videos en donde el rostro tenga mayor movilidad.

Para presentar el primer conjunto de datos, se tomaron videos de la misma persona en diferentes escenarios, en ninguno de ellos la persona involucrada realiza muchas expresiones o movimientos del rostro o cuerpo. La Figuras 4.41a y 4.41b muestran el conjunto de

datos utilizado y la Figura 4.42 muestra el resultado[29] en donde el rostro se ven bien formado y con la correcta posición de sus atributos. Generar estos *deepfakes* es muy sencillo ya que basta editar el rostro en los primeros fotogramas y comprobar que el resultado sea el deseado, ya que la cara mantiene la misma posición el resto del video.

(a) Rostro fuente. (b) Rostro destino.

Figura 4.41: Conjunto de datos, rostros con poca movilidad.

Figura 4.42: *deepfake* con poca movilidad.

Por otro lado, se realizaron experimentos en donde el rostro tuviera más movilidad y buscando que ambos videos, fuente y destino, contuvieran ángulos y movimientos similares para que se obtuvieran mejores resultados, es decir si el rostro del individuo destino se encontraba viendo hacia la derecha, se buscaba un video fuente que contuviera esos mismos ángulos. Los rostros obtenidos se ven más reales y claros. La Figura 4.43 muestra

[29] https://youtu.be/DgA_KU1AVU8

el conjunto de datos utilizado. Ambas personas se encuentran viendo hacía la misma dirección, para que el sistema pudiera aprender esos lados. La Figura 4.44 muestra el resultado[30]. El cuerpo y el rostro presentan movimientos hacia el lado izquierdo y derecho, de modo que se buscó un rostro que tuviera movimientos similares para reemplazar. Este tipo de *deepfakes* presentan un mayor reto a la hora de unificar el rostro aprendido con el destino, ya que se tiene que trabajar con las orillas del rostro para que no sean visibles en algunos ángulos. En este caso, se alcanza a apreciar del lado superior derecho en algunos cuadros del video, una línea que divide ambos rostros; algunos detalles no se alcanzan a percibir a simple vista o las primeras veces y puede que con el movimiento del video se pierdan, sin embargo si se presta atención pueden ser los que revelen la falsedad.

(a) Rostro fuente.

(b) Rostro destino.

Figura 4.43: Conjunto de datos, rostros con poca movilidad.

Figura 4.44: *deepfake* con mayor movilidad.

[30]https://youtu.be/PQUIcqqlT9w

Ensayo:Videos domésticos con diferente movilidad del rostro.

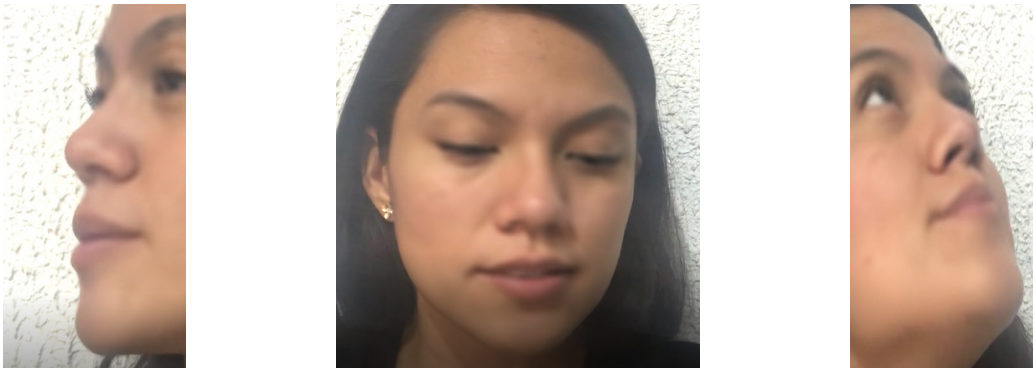
Hasta el momento se sabía que no tener conjuntos de datos con movimientos semejantes afecta el resultado, sin embargo se decidió grabar dos rondas de videos domésticos para estimar los puntos débiles de esta característica. En la primera ronda se grabaron videos en donde el rostro gira aproximadamente 80° grados a la derecha y hacia la izquierda y cuando regresa al centro realiza movimientos hacia arriba y hacia abajo. En la segunda ronda se realiza una toma del rostro más completa, empieza su trayectoria alzando la mirada y el rostro hacia algún extremo (izquierdo o derecho) y comienza a girarlo hasta el otro, como si se estuviera viendo pasar un avión. Cuando termina con este movimiento, realiza dos recorridos más, uno girando el rostro en la parte media (como si se viera pasar una persona de un lado a otro) y el otro con la cara y la mirada viendo hacia la parte inferior (como si se viera caminar un insecto). Ambos experimentos se realizaron con el mismo conjunto de datos y después intercambiando el rostro de diferentes personas, los *deepfake* obtenidos sobre ellos mismos no presentan fallos, se acoplan correctamente con los giros y movimientos de la cara como se puede ver en la Figuras 4.45 y 4.47. Sin embargo, con los rostros de otras personas se tiene fallos regularmente cuando el rostro hace un giro con más de 60° o rostros muy inclinados como los que se muestran en la Figuras 4.46 y 4.48. En ellos se empiezan a notar los dos rostros y rostros muy distorsionados. La Figura 4.48b muestra un ejemplo de un rostro con una inclinación menor a 60° en donde todavía se puede ver la cara con una buena forma y correcta posición de sus atributos. Por otro lado, en las Figuras 4.48a y 4.48c, que tienen mayor inclinación, se puede ver el rostro movido e incompleto.

(a) *deepfake* lado derecho. (b) Inclinación hacia arriba. (c) *deepfake* lado izquierdo.

Figura 4.45: Primer video doméstico con poca movilidad - mismo conjunto datos.

(a) *deepfake* lado derecho. (b) Inclinación hacia arriba. (c) *deepfake* lado izquierdo.

Figura 4.46: Primer video doméstico con poca movilidad - diferente conjunto datos.



(a) *deepfake* lado derecho. (b) Inclinación hacia abajo. (c) *deepfake* lado izquierdo.

Figura 4.47: Segundo video doméstico con mayor movilidad - mismo conjunto datos.

(a) *deepfake* lado derecho ($< 60°$). (b) *deepfake* con ángulo $>60°$. (c) *deepfake* lado izquierdo ($< 60°$).

Figura 4.48: Segundo video doméstico con mayor movilidad - diferente conjunto datos.

4.4. Ajuste de los Hiperparámetros

4.4.1. Tiempo de entrenamiento

Esta característica está relacionada con el software generador de *deepfakes*, es decir no es una característica que se encuentre dentro de un video, sin embargo es importante mencionarla de manera más extensa ya que la obtención de buenos resultados va de la mano con el tiempo que se entrene el modelo.

Se decidió llevar un control sobre el tiempo de entrenamiento para observar los avances que el *deepfake* tenía y estimar con cuántas horas de entrenamiento se podía lograr un resultado satisfactorio. Para ello se estableció un tiempo total de 8 horas, realizando pausas cada dos horas. A continuación se mencionan algunas situaciones que pueden encontrarse.

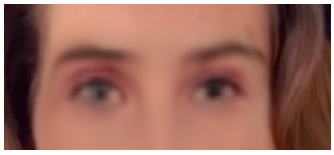
4.4.2. Estudio inicial.

El estudio inicial se realizó a las dos horas de entrenamiento, donde se puede tener una evaluación preliminar con aspectos como: si el rostro fuente va a estar bien posicionado o se encuentra movido, también si el rostro resultante iguala el tono de piel al destino o si en la gama de colores se encuentra un tono de piel más adecuado. También se puede ver si contiene manchas sobre la cara o en los alrededores y si es que pueden ser difuminadas, además, al crear el *deepfake* se verá la movilidad o la coordinación de los movimientos de los labios respecto al audio, ya que hay casos en los que los labios no se mueven como deberían. Son pocas las cosas que se pueden notar, sin embargo tienen mucho peso para saber si vale la pena continuar con el entrenamiento o se debería cambiar el conjunto de datos.

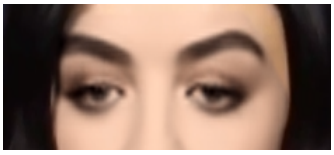
4.4.3. Resultados con pequeños defectos.

Dentro de las 4 a 6 horas de entrenamiento todavía hay rostros que no se encuentran bien definidos, inclusive hay movimientos que pueden alcanzarse a ver un poco borrosos si se hace una pausa en el video. Un fallo que se encontró particularmente en varios videos es el color de ojos y los movimientos de éstos. La Figura 4.49a muestra un ejemplo en donde el ojo derecho tiene un color más claro que el otro. En otros casos la mirada se encuentra perdida, o como si el individuo se encontrara en un estado de cansancio, ver en

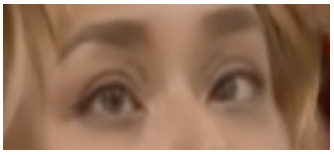
la Figura 4.49b. Finalmente, se encontraron situaciones en la que un ojo está viendo para un lado distinto como en la Figura 4.49c.



(a) Ojos de diferente color.



(b) Ojos cansados.



(c) Distinta dirección.

Figura 4.49: Defectos en los ojos.

Otros detalles comunes que pueden apreciarse son que los dientes: nunca están bien definidos. Independientemente de que la calidad baja cuando se genera el *deepfake*, como se ha visto en los casos que se utilizó el mismo video como fuente y destino. Los videos originales tienen una buena calidad. Sin embargo, al realizar todo el procedimiento para la generación de *deepfakes* ésta disminuyó. A pesar del tiempo de entrenamiento, los dientes se verán borrosos como se muestra en la Figura 4.50.

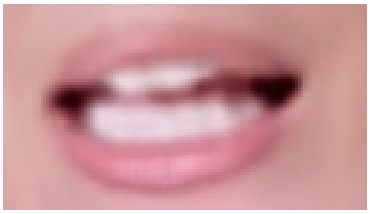
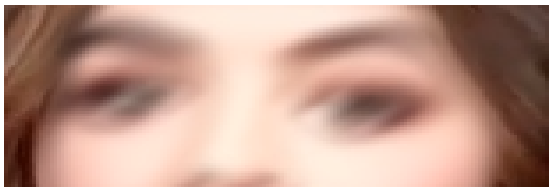


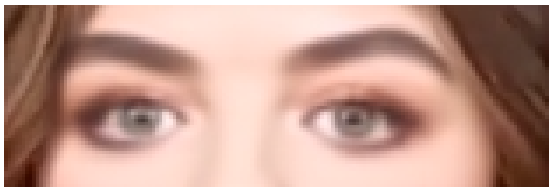
Figura 4.50: Defectos en los dientes.

Tiempo mínimo idóneo

Se genera un resultado positivo con al menos 8 horas de entrenamiento. Este tiempo no asegura obtener el mejor resultado sin embargo, es considerado un buen tiempo para solucionar los problemas anteriores. En este tiempo se presenta una mayor estabilidad en los ojos, y se encontraron mejoras en tomas borrosas en comparación con un menor tiempo de entrenamiento, como se muestra en la Figura 4.51. Asimismo mejora el movimiento de los labios y la textura del rostro se ve más lisa.



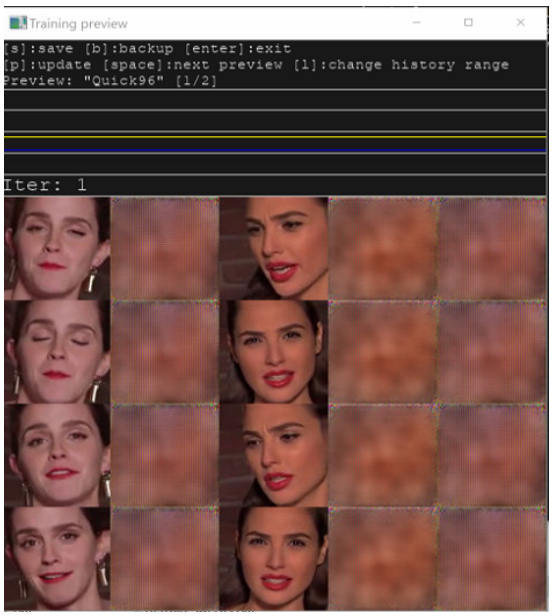
(a) 2 horas de entrenamiento.



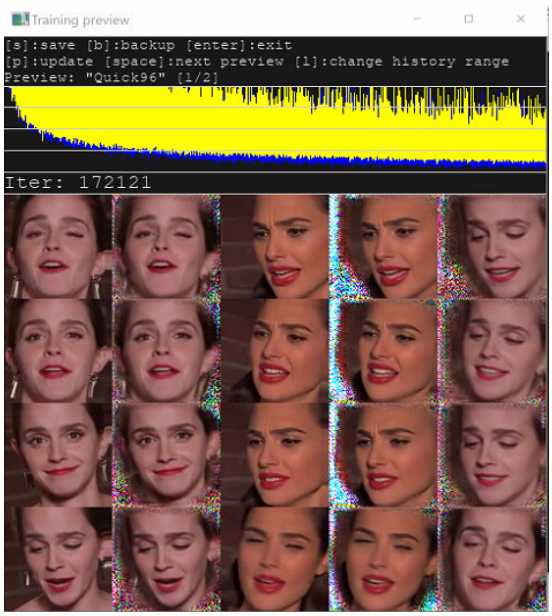
(b) 8 horas de entrenamiento.

Figura 4.51: Rostros borrosos.

Esta es solo una estimación y descripción breve de algunos fallos que se presentaron en esta experimentación dentro del tiempo de entrenamiento. Si el usuario decide entrenar el modelo en un tiempo mayor puede tomar en cuenta lo siguiente. Al momento de realizar el entrenamiento *DeepFaceLab* ofrece una vista donde se observa el progreso de entrenamiento como se muestra en la Figura 4.52. Esta vista se irá actualizando conforme el modelo vaya aprendiendo el rostro, si esta vista no se actualiza puede presionar la tecla **p**. Con vista actualizada el usuario considerará detener el entrenamiento si los rostros se ven claros o si ya no se ven borrosos como en la Figura 4.52a.



(a) 1 iteración



(b) 172 mil iteraciones.

Figura 4.52: Visualización del entrenamiento.

Por otro lado, también se puede ver el desempeño del entrenamiento en la consola del sistema como se muestra en la Figura 4.53. En ella se pueden encontrar los siguientes valores de izquierda a derecha. El primer valor es la hora actual, este muestra la hora a

la que empieza el entrenamiento y se guardarán los valores cada 15 minutos. El segundo valor es el número de iteración en la que se encuentra. El tercer valor representa el tiempo de iteración. El cuarto valor es el costo *loss* o de pérdida para el rostro fuente y el último el valor *loss* o pérdida del rostro destino.

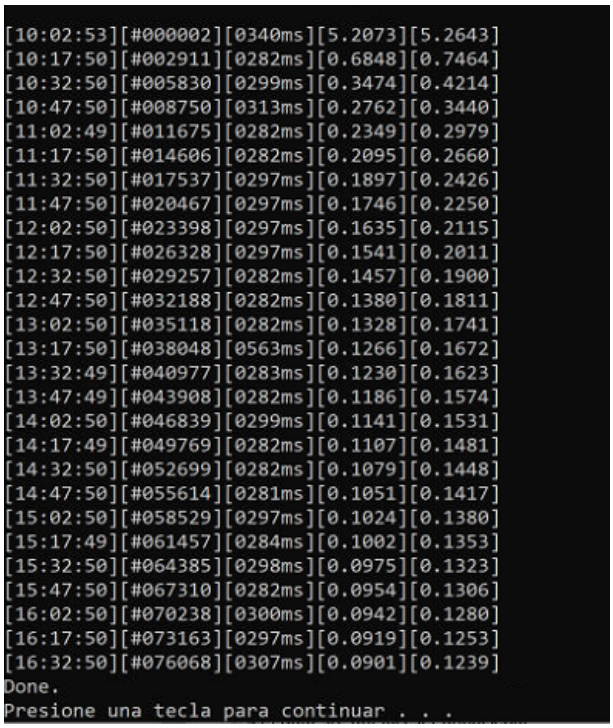


Figura 4.53: Valores de entrenamiento en consola.

En la Figura 4.54 se pueden ver más a detalle estos valores. Como se ha mencionado anteriormente, se encontró que el tiempo de 8 horas de entrenamiento puede ser un tiempo idóneo para obtener un buen resultado. Si se desea contabilizar el tiempo de entrenamiento, el valor de la hora actual sirve para conocer el tiempo total que lleva el modelo en entrenamiento. Otro momento adecuado para detener el entrenamiento es cuando los costos de pérdida no varían o varían muy poco, de un renglón a otro, ya que el modelo puede no estar aprendiendo más. Sin embargo, no hay una respuesta definitiva sobre cuando se deba parar el entrenamiento, ya que puede variar según el conjunto de datos. Se recomienda revisar el resultado en el tiempo idóneo mencionado y si el usuario considera que se puede obtener un mejor resultado podría extender el entrenamiento.

Figura 4.54: Valores de entrenamiento a detalle.

4.4.4. Hiperparámetros de la red neuronal

Se utilizó el sistema deepfake en [12] para realizar cambios sobre los hiperparámetros de la red neuronal y analizar si realizando modificaciones sobre la red se obtienen mejores resultados. En primer lugar, se cambió el optimizador original Adam por los optimizadores Adamax, Nadam, RMSprop, Adadelta y Adagrad. Estos dos últimos no dieron resultados favorables por lo cual fueron descartados y sólo se trabajó con los primeros 4.

Para realizar estas pruebas se entrenaron 5 videos con cada optimizador con 60 mil iteraciones y se guardaron sus tiempos de entrenamiento y valores de pérdida.

La Tabla 4.8 muestra los valores de pérdida en la epoch 60,000 de los optimizadores probados. Comparando los valores de pérdida, se mostró que en los 5 videos los optimizadores que obtenian el mejor costo fueron Adam y Nadam.

Tabla 4.8: Valores de pérdida epoch 60,000.

Optimizador	Video 1	Video 2	Video 3	Video 4	Video 5
Adam	**.01646209**	0.0107654	0.02224902	0.01434028	**.01054084**
Adamax	0.02150065	0.01464975	0.02427447	**.00996466**	0.01402692
Nadam	0.01695776	**0.01047028**	**.01690554**	0.01279657	0.01068865
RMSprop	0.01778553	0.01140416	0.01902099	0.01329578	0.01091726

Por lo que se decidió continuar los experimentos solo con estos dos optimizadores. Las siguientes pruebas consistieron en modificar el número de capas con las que se trabajaba. Para ello, se agregó y quitó una capa en el codificador y decodificador. En la Tabla 4.9 se presentan los valores de pérdida del optimizador Adam. Dentro de los paréntesis se encuentra el número de capas que se tienen en el codificador y el decodificador. Se puede ver que el modelo Adam con 5 capas obtuvo un mejor costo en la epoch 60,000.

Tabla 4.9: Valores de pérdida epoch 60,000 optimizador Adam.

Optimizador	Video 1	Video 2	Video 3	Video 4	Video 5
Adam (3)	0.017389826	0.011261582	0.01580286	0.012970004	0.01026934
Adam (4)	**0.01646209**	0.0107654	0.02224902	0.01434028	0.01054084
Adam (5)	0.016861835	**.009865493**	**.015246582**	**.011253656**	**.008860294**

Por otro lado, en la Tabla 4.10 se presentan los valores de pérdida del optimizador Nadam. En ella se muestra que el modelo con 4 capas y 5 capas obtuvieron los mejores resultados. A diferencia del optimizador anterior, en este caso es difícil escoger la mejor opción debido a los ejemplos seleccionados.

Tabla 4.10: Valores de pérdida epoch 60,000 optimizador Nadam.

Optimizador	Video 1	Video 2	Video 3	Video 4	Video 5
Nadam (3)	0.02447663	0.01601651	0.020289343	0.019088766	0.014959887
Nadam (4)	**.01695776**	0.01047028	**.01690554**	0.01279657	**.01068865**
Nadam (5)	0.01766818	**.008475093**	0.016944516	**.010114445**	0.010962083

Para obtener un resumen de los mejores costos obtenidos entre los modelos Adam y Nadam en la epoch 60,000 se presenta la Tabla 4.11.

Tabla 4.11: Valores de pérdida epoch 60,000 optimizadores Adam y Nadam.

Optimizador	Video 1	Video 2	Video 3	Video 4	Video 5
Adam (5)	**.016861835**	0.009865493	**.015246582**	0.011253656	**.008860294**
Nadam (4)	0.01695776	0.01047028	0.01690554	0.01279657	0.01068865
Nadam (5)	0.01766818	**.008475093**	0.016944516	**.010114445**	0.010962083

Finalmente, se escogió el optimizador que tuvo mejor resultado, Adam, y se decidió incrementar una capa más. Estos costos son presentados en la Tabla 4.12.

Tabla 4.12: Valores de pérdida epoch 60,000 adam 6 capas.

Optimizador	Video 1	Video 2	Video 3	Video 4	Video 5
Adam (5)	0.016861835	0.009865493	0.015246582	0.011253656	0.008860294
Adam (6)	0.022894064	0.014874776	0.019928833	0.015378582	0.014028397

El modelo de 6 capas no obtuvo mejores costos que el modelo de 5 capas. Sin embargo, el modelo de 5 capas presentó mejores costos que el modelo original, además de presentar tiempos de cómputo menores.

Capítulo 5

Conclusiones

En este trabajo de investigación se mencionaron las características que se consideraron influyentes en el resultado y algunos de los problemas con los que el usuario se puede encontrar al realizar *deepfakes*. A continuación se presentan recomendaciones idóneas sobre las características a seguir para obtener un buen resultado.

Iluminación. Esta característica es muy influyente en el resultado, se percató de que cuando la iluminación de los videos de entrada no es similar, en el *deepfake* resultante se pueden mostrar manchas brillosas o sombras sobre el rostro, líneas blancas o negras en los extremos, o una máscara muy marcada en el contorno. De hecho, aunque el tono de piel de los involucrados en los videos originales sea similar, si la iluminación es distinta puede mostrar rostros con tono de piel contrastante, incluso se puede apreciar la diferencia entre la textura de piel. Los experimentos realizados prueban que teniendo iluminación similar entre videos, los resultados no presentan este tipo de anomalías. Por lo que se recomienda que el conjunto de datos tenga iluminación similar y uniforme, es decir, que los individuos no presenten sombras en el rostro ya que esto puede afectar el resultado.

Tamaño del rostro / Diferente encuadre. Seleccionar videos con diferente encuadre no es un factor muy influyente en el resultado. Algunos de los problemas que se enfrentaron experimentando con rostros de diferente tamaño fue ajustarlos al rostro destino, pero *DeepFaceLab* hace que el rostro resultante tenga el tamaño adecuado. En caso de considerar que no es el apropiado se puede editar difuminando y erosionando los bordes hasta lograr un buen resultado o bien, se puede hacer más grande y pequeño sin ninguna dificultad, sin embargo, se debe de revisar en el *deepfake* final que los atributos del rostro embonen y muestren un rostro natural. Por otro lado, no se recomienda trabajar *close up*

shots ya que la toma hacia el rostro es muy amplia y se puede apreciar a detalle todos los defectos y fallas que no se solucionen, como la diferencia entre los rostros, además de que en los resultados se encontraron rostros borrosos y mal definidos. Los atributos como barba y ceja suelen verse muy difuminados, lo que hace que el resultado no se vea realista. Lo ideal sería trabajar con *medium shots*, esto es con MCU y MS, ambos son buenos para lograr el objetivo de engañar a los que vean el *deepfake* debido a que, independientemente del dispositivo electrónico en el que se vea, es más difícil que se note alguna irregularidad en un rostro pequeño o alejado y sus atributos se muestran más naturales.

Afinidad de rostros. En los experimentos realizados se mostró la importancia de considerar la afinidad de las caras, en una de las pruebas un rostro fuente con forma de diamante generaba problemas en un rostro más ovalado. El algoritmo creado funcionó muy bien para mostrar las coordenadas entre el contorno del rostro y los atributos del conjunto de datos y ayudó a generar una matriz de similitud entre personajes, lo cual expusó que si se realizaban *deepfakes* con personas que tuvieran atributos con mayor semejanza, el rostro fuente puede acoplarse mejor en el rostro destino.

Tono de piel. Trabajar con la tonalidad de piel puede ser una tarea muy fácil o muy difícil. El sistema *DeepFaceLab* trata de igualar el tono de piel del rostro fuente al del rostro destino, cuando no se iguala con éxito se puede probar con la gama de colores que cuenta el sistema y seleccionar la mejor opción. Sin embargo, esta tarea no siempre es exitosa. Para ello se probó entre los dos modelos de entrenamiento Quick96 y SAEHD para ver su desempeño. El modelo SAEHD tiene un mejor comportamiento igualando los tonos, pero en algunos casos todavía es notorio el contorno de la máscara en la parte frontal y lateral del rostro. Por otro lado, en los experimentos en los que se modificó el tono de piel, se percató que realizar cambios en éstos no es muy conveniente ya que los resultados no son significativos. Además que realizar el preprocesamiento de todos estos datos es un trabajo complicado. Se recomienda utilizar conjuntos de datos en los cuales los individuos tengan un tono de piel muy semejante, ya que será más fácil trabajar con ellos sin necesidad de hacer muchas modificaciones, además de que el resultado ocultará la linea de la máscara y la división entre rostros mostrando un producto más natural.

Movimiento y rotación del rostro en distintos ángulos. Esta característica es muy importante y de las más influyentes en el resultado. Para generar buenos *deepfakes* es necesario conseguir conjuntos de datos en donde los individuos presenten los mismos

ángulos y rotaciones, es decir, si la persona esta haciendo movimientos de su rostro hacia la izquierda o derecha, no se obtendrá un buen resultado si la persona del otro video se encuentra viendo solamente hacia enfrente. Además, las pruebas caseras mostraron que se puede obtener un resultado creíble cuando el rostro no gira más de aproximadamente 60°, ya que cuando el rostro realiza estas rotaciones y no se aprendió correctamente la cara del individuo fuente, se pueden ver rostros deformados o inclusive se pueden ver ambos rostros. En la web se han encontrado diferentes ejemplos de *deepfakes* en los que cuando una persona se encuentra posicionada completamente de perfil el deepfake se pierde y se muestra el rostro original.

Tiempo de entrenamiento. El tiempo idóneo encontrado en este trabajo de investigación para los ejemplos realizados fue de 8 horas. En este tiempo se podían obtener resultados exitosos. En tiempos menores se presentaban errores en: ojos, dientes y rostro como: ojos de diferente color, o bien ojos que veían para lados diferentes. Sin embargo, como se mencionó en la subsección de **tiempo de entrenamiento**, no se sabe con exactitud el mejor momento para detener esta tarea, el usuario puede guiarse por los valores de pérdida. Por otro lado, para evitar entrenamientos muy largos y malos resultados, se recomienda entrenar el conjunto de videos por un tiempo corto y realizar las modificaciones necesarias para conocer si se podrá obtener un buen resultado con mayor número de iteraciones.

Hiperparámetros de la red neuronal. Los ajustes realizados en los hiperparámetros de la red neuronal trajeron mejorías sobre los costos y tiempos computacionales. Visualmente también se observa una mejoría en los resultados obtenidos entre el modelo original de 4 capas y los modelos con mayor número de capas.

Todos los resultados de los experimentos mostraron una mejoría sobre los resultados obtenidos. Se recomienda seguirlas para lograr buenos *deepfakes*. Sin embargo, también estos experimentos buenos tanto como fallidos sirvieron para identificar en qué parte de los videos se debe poner más atención para buscar algún error y no ser engañados tan fácilmente.

Bibliografía

[1] R. Tolosana, R. Vera-Rodriguez, J. Fierrez, A. Morales, and J. Ortega-Garcia, "Deepfakes and beyond: A survey of face manipulation and fake detection," *Information Fusion*, vol. 64, pp. 131–148, 2020.

[2] N. C. Köbis, B. Doležalová, and I. Soraperra, "Fooled twice: People cannot detect deepfakes but think they can," *Iscience*, vol. 24, no. 11, p. 103364, 2021.

[3] A. Lewis, P. Vu, A. Chowdhury, *et al.*, "Do content warnings help people spot a deepfake? evidence from two experiments," 2022.

[4] M. Masood, M. Nawaz, K. M. Malik, A. Javed, and A. Irtaza, "Deepfakes generation and detection: State-of-the-art, open challenges, countermeasures, and way forward," *arXiv preprint arXiv:2103.00484*, 2021.

[5] Y. Nirkin, Y. Keller, and T. Hassner, "Fsgan: Subject agnostic face swapping and reenactment," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 7184–7193, 2019.

[6] R. Chen, X. Chen, B. Ni, and Y. Ge, "Simswap: An efficient framework for high fidelity face swapping," in *Proceedings of the 28th ACM International Conference on Multimedia*, pp. 2003–2011, 2020.

[7] L. Li, J. Bao, H. Yang, D. Chen, and F. Wen, "Faceshifter: Towards high fidelity and occlusion aware face swapping," *arXiv preprint arXiv:1912.13457*, 2019.

[8] I. Perov, D. Gao, N. Chervoniy, K. Liu, S. Marangonda, C. Umé, M. Dpfks, C. S. Facenheim, L. RP, J. Jiang, *et al.*, "Deepfacelab: A simple, flexible and extensible face swapping framework," *arXiv preprint arXiv:2005.05535*, 2020.

[9] D. Bank, N. Koenigstein, and R. Giryes, "Autoencoders," *arXiv preprint arXiv:2003.05991*, 2020.

[10] M. Albahar and J. Almalki, "Deepfakes: Threats and countermeasures systematic review," *Journal of Theoretical and Applied Information Technology*, vol. 97, no. 22, pp. 3242–3250, 2019.

[11] Q. Galvane, *Automatic Cinematography and Editing in Virtual Environments.* PhD thesis, Université Grenoble Alpes (ComUE), 2015.

[12] llSourcell, "Github - llsourcell/deepfakes: This is the code for "deepfakes" by siraj raval on youtube," Feb 2018.

[13] B. Yegnanarayana, *Artificial neurals networks.* PHI Learning Pvt. Ltd.yegnanarayana, 2009.

[14] F. Marini, R. Bucci, A. Magrì, and A. Magrì, "Artificial neural networks in chemometrics: History, examples and perspectives," *Microchemical journal*, vol. 88, no. 2, pp. 178–185, 2008.

[15] N. Buduma and N. Locascio, *Fundamentals of deep learning: Designing next-generation machine intelligence algorithms.* .ºReilly Media, Inc.", 2017.

[16] I. A. Basheer and M. Hajmeer, "Artificial neural networks: fundamentals, computing, design, and application," *Journal of microbiological methods*, vol. 43, no. 1, pp. 3–31, 2000.

[17] J. Krohn, G. Beyleveld, and A. Bassens, *Deep learning illustrated: A visual, interactive guide to artificial intelligence.* Addison-Wesley Professional, 2019.

[18] A. Géron, *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems.* O'Reilly Media, 2019.

[19] C. C. Aggarwal *et al.*, "Neural networks and deep learning," *Springer*, vol. 10, pp. 978–3, 2018.

[20] F. Chollet, *Deep learning with Python.* Simon and Schuster, 2021.

[21] S. Skansi, *Introduction to Deep Learning: from logical calculus to artificial intelligence*. Springer, 2018.

[22] S. Sharma, S. Sharma, and A. Athaiya, "Activation functions in neural networks," *towards data science*, vol. 6, no. 12, pp. 310–316, 2017.

[23] A. F. Gad and F. E. Jarmouni, *Introduction to Deep Learning and Neural Networks with PythonTM: A Practical Guide*. Academic Press, 2020.

[24] B. Karlik and A. V. Olgac, "Performance analysis of various activation functions in generalized mlp architectures of neural networks," *International Journal of Artificial Intelligence and Expert Systems*, vol. 1, no. 4, pp. 111–122, 2011.

[25] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

Apéndice A

Redes Neuronales Artificiales

A.1. Historia

La historia de las redes neuronales artificiales empezó por el interés de simular la actividad neuronal de forma computacional. El primer modelo para su desarrollo fue propuesto en 1943 por Warren McCulloch y Walter Pitts llamado "neurona McCulloch-Pitts", como su nombre lo menciona fue un modelo de una sola neurona el cual tomaba el número de entradas dadas y cada una la asociaba con un número de pesos para realizar lo que se conoce como suma ponderada, en algunos casos se incluían sesgos y la resultante pasaba por una función de activación para producir la salida y transmitirla a otras neuronas. Sin embargo, este modelo tuvo un obstáculo, ya que el valor asignado de los pesos era fijo y por lo tanto no permitía realizar un aprendizaje sobre los ejemplos dados [13] [14] [15]. En 1949 Donald Hebb lanzó una propuesta en donde se permitía realizar el ajuste en estos pesos basado en los valores pre y postsinápticos, y en 1953 Marvin Minsky desarrolló una máquina de aprendizaje la cual permitía que los valores de los pesos se adaptaran de forma automática. Pero fue hasta el año de 1958, con las bases del modelo generado por McCulloch y Pitts, en el que Rosenblatt desarrolló una red basada en una sola unidad conocida como perceptrón con pesos ajustables, la cual demostró ser eficiente para la resolución de problemas de clasificación lineal. En 1962 Rossenblatt publicó un libro llamado "Principios de la Neurodinámica", sin embargo en 1969 Minsky y Papert mostraron que el modelo que presentaba Rosenblatt contaba con ciertas limitaciones que no se podían superar con agregar múltiples capas de neuronas, una de éstas era que el modelo no era útil para la resolución de problemas de clasificación no lineales, lo que puso en

pausa el desarrollo de modelos neuronales hasta 1984, este período fue conocido como los años tranquilos y la investigación silenciosa [13] [14] [16]. El interés hacia el área de la inteligencia artificial volvió a principios de los años 80's, debido a Hopfield y su análisis a las redes en 1982, en donde mostró la existencia de estados de equilibrio estables en este tipo de red, y Rumelhart en 1986 quien redescubrió el algoritmo de aprendizaje por retropropagación, el cual había sido desarrollado por Werbos en 1974 [13] [16]. Hoy en día las aportaciones que han traído consigo las redes neuronales han sido grandes, al igual que los campos en los que se han aplicado para la resolución de diversos problemas. En las siguientes secciones de este apéndice se profundiza en algunos de los temas más relevantes mencionados en esta breve historia de las redes neuronales.

A.2. La neurona

Como se mencionó en la sección anterior, las redes neuronales se basan en el funcionamiento de una neurona. La anatomía de ésta se muestra en la Figura A.1 y está compuesta por elementos que representan las neuronas artificiales, entre ellos se encuentran: las dendritas, los axones y la transmisión de señales de una neurona a otra llamada sinapsis. Las dendritas están conectadas al cuerpo celular, en donde se encuentra localizado el núcleo y se extienden por una larga fibra llamada axón, al final los axones están conectados a otras neuronas a través de terminales sinápticas y reciben señales mediante la sinapsis activando a la neurona y así sucesivamente con las demás.

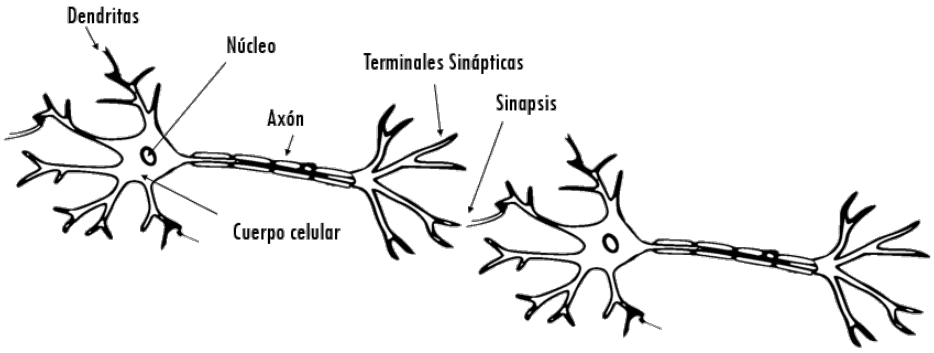
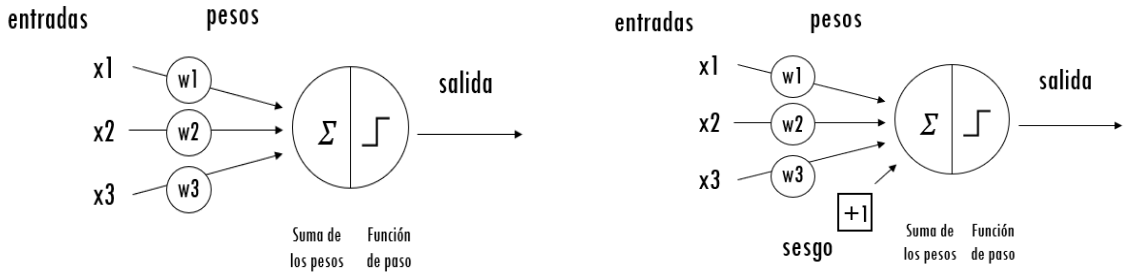


Figura A.1: Anatomía de una neurona

A.3. Perceptrón

A finales de los años 1950, el neurobiólogo Frank Rossenblatt creó la red neuronal más simple que se conoce, el perceptrón, influenciado por su comprehensión de las neuronas biológicas. Básicamente consiste en una neurona, basada en una neurona artificial llamada unidad lógica de umbral TLU (threshold linear unit) en donde las entradas son números en lugar de valores binarios y cada entrada está asociada con un peso como se muestra en la Figura A.2a y en algunos casos incluye una neurona de sesgo como se ve en la Figura A.2b. La unidad lógica de umbral calcula una suma ponderada de sus entradas y luego aplica una función de paso o función escalonada para generar una salida [17] [18].



(a) Perceptrón sin sesgo

(b) Perceptrón con sesgo

Figura A.2: Arquitectura básica del perceptrón

El modelo del perceptrón puede ser utilizado para una clasificación lineal binaria simple a conjuntos de datos como el que se puede ver en la Figura A.3a, se demostró que con este tipo de datos el algoritmo perceptrón siempre converge, sin embargo, cuando se tiene un conjunto de datos no linealmente separable, como en la Figura A.3b, tiende a mostrar un desempeño deficiente y no se garantiza su convergencia [19].

(a) Linealmente separables

(b) Linealmente inseparables

Figura A.3: Ejemplo de conjuntos separables e inseparables en dos clases

A.3.1. Perceptrón Multicapa

El modelo simple del perceptrón presentó algunas limitaciones las cuales no pueden resolverse apilando múltiples perceptrones, un ejemplo de estas limitaciones es el problema del XOR. El resultado del apilamiento de perceptrones es una red neuronal artificial conocida como perceptrón multicapa (*Multilayer Perceptron*, MLP) [18], se compone de múltiples capas, la capa de entrada, la capa de salida y las capas intermedias a estas son conocidas como capas ocultas, además a esta arquitectura se le denomina redes de alimentación directa (feedforward networks) debido a que las capas sucesivas se alimentan entre si en la dirección de avance comenzando desde la entrada hasta la salida. Esta arquitectura asume que todos las neuronas en una capa están conectadas a la siguiente, la Figura A.4 muestra un ejemplo de esta red. Para la creación de este tipo de redes se necesitan ciertos parámetros los cuales se verán con más detalle a continuación, como lo son el número de capas que tendrá la red, asimismo el número de neuronas y la función de activación.

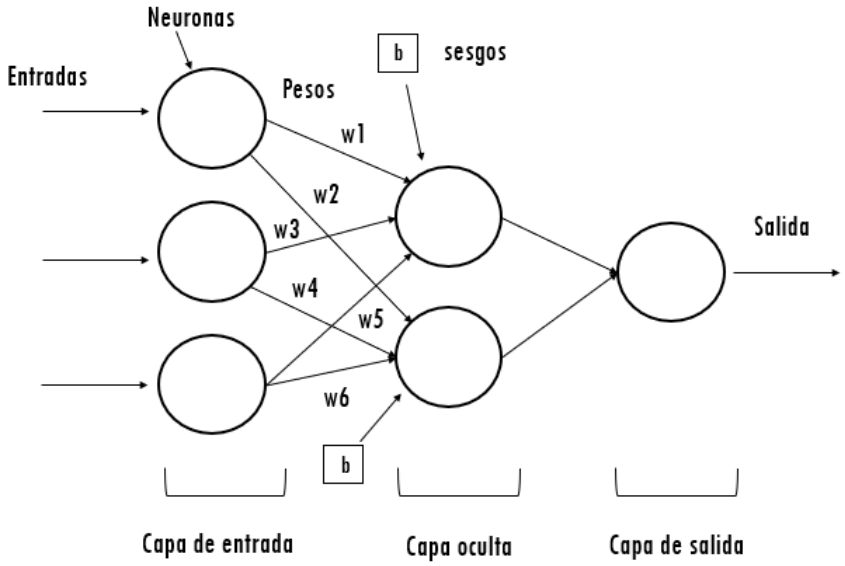


Figura A.4: Arquitectura de una red neuronal artificial (ANN)

Arquitectura

A continuación, se explican con mayor detalle cada uno de los elementos que conforman una red neuronal multicapa:

Capa de entrada

Es la primera capa de la red, se encuentran las neuronas también conocidas como unidades básicas, el número de neuronas que se encontrarán en esta capa corresponderá al número de variables de entrada.

Capa oculta

Comúnmente una red neuronal cuenta con varias de estas capas, pueden ser predecesoras de la capa de entrada o bien de alguna otra capa oculta, las neuronas de la capa de entrada están conectadas a las neuronas de las capas ocultas por pesos y sesgos, que permiten realizar una serie de cálculos sobre los datos de entrada que pasarán a la capa de salida. Tener más capas ocultas permite que la red aprenda representaciones más complejas, pero hace que el costo de la red sea computacionalmente mayor, inclusive conduce a patrones de aprendizaje no deseados, por eso es importante saber cuántas capas se deben utilizar, así como el número de neuronas que cada una de estas tendrá [20].

Capa de salida

Es la última capa de la red, por ende, sucesora a la última capa oculta, el número de neuronas en esta capa depende del número de clases distintas en la tarea de clasificación [13], al igual que la capa anterior se realizarán una serie de cálculos sobre los datos recibidos para poder generar el resultado a los datos de entrada.

Pesos

Las neuronas de cada capa son conectadas a todas las neuronas de la siguiente capa mediante pesos, por lo tanto el número total de pesos estará dado por el número de neuronas de la capa antecesora, multiplicado por el número de neuronas en la siguiente capa. Son representados por la letra w por su nombre en inglés *(weight)* y su función es controlar la cantidad de características que se deja pasar a la siguiente capa, o bien que es transmitida a la siguiente. En la implementación los pesos pueden ser inicializados a cualquier valor aleatorio [13]. Regularmente estos valores se encuentran entre el intervalo de 0 y 1, para pesos superiores a 1 se piensa en ellos como una amplificación [21]. Una vez asignados los pesos a cada unidad básica el procedimiento que se realiza con ellos se conoce como suma ponderada (la suma de los pesos multiplicados por sus valores de entrada) y después se aplicará una función de activación para convertir el valor agregado en una etiqueta de clase. Asimismo, estos pesos más adelante se ajustarán en respuesta a errores de predicción, con el objetivo de hacer las predicciones más correctas en iteraciones futuras. Los pesos se cambiarán cuidadosamente para reducir el error del cálculo [19].

Sesgos

También conocido por *bias* por su nombre en inglés y denotado por la letra b, históricamente se le ha llamado *umbral* o *threshold*, y tiene un comportamiento diferente al peso. Aunque también puede ser generado de manera aleatoria [21]. El sesgo puede ser incorporado a la red utilizando una neurona de sesgo, el número de estas neuronas será igual al número de neuronas que se tenga en las capas ocultas o en la capa de salida [19].

Función de activación

Los valores de entrada de una red neuronal pasan por ciertos cálculos antes de ser enviados a la capa siguiente. Como se mencionó anteriormente, este cálculo consiste en realizar la suma de los productos de las entradas por sus pesos correspondientes, y si es el caso en el que la red neuronal cuente con sesgos, estos son sumados igualmente. Ésto puede ser representado por la ecuación A.1.

$$Output = sesgos + \sum(pesos * valores\ de\ entrada) \tag{A.1}$$

Pero al resultado todavía debe aplicarse lo que se conoce como función de activación, para finalmente transformarse en la señal de salida. Si la función de activación no se usa, la salida sería simplemente una función lineal y no tiene la capacidad de aprender y reconocer asignaciones complejas a partir de datos, es decir, el modelo lineal sólo representa la relación entre las entradas y salidas como una línea, pero desafortunadamente los datos de muchos problemas no pueden modelarse de forma lineal. Es por eso que debe crearse un modelo no lineal para reducir el error y se logra aplicando una función de activación no lineal. La precisión de una red neuronal es definida por el tipo de función de activación que se utilice [22] [23]. Existen diferentes tipos de funciones de activación. Entre las funciones lineales se encuentran: la función de paso binario y la función lineal. Entre las no lineales se encuentran: la función Sigmoide, función Tangenta hiperbólica, función ReLU, función Leaky ReLU, función ReLU parametrizada y Softmax.

Función Sigmoide

La función de activación Sigmoide transforma el resultado obtenido por la ecuación A.1, este valor también es conocido como valor de pre-activación, cuando la función de activación se aplica da como resultado el valor de salida o valor post-activación entre un rango de 0 a 1, el cual es útil para realizar cálculos que se puedan interpretar como probabilidades [19] [22], es muy común que se utilice esta función en redes neuronales

entrenadas por retropropagación, ya que es muy fácil de distinguir y puede minimizar la capacidad del cálculo de entrenamiento [24]. La ecuación que la define está dada por A.2 y tiene una representación gráfica como se muestra en la Figura A.5.

$$f(x) = \frac{1}{1 + e^{x}} \tag{A.2}$$

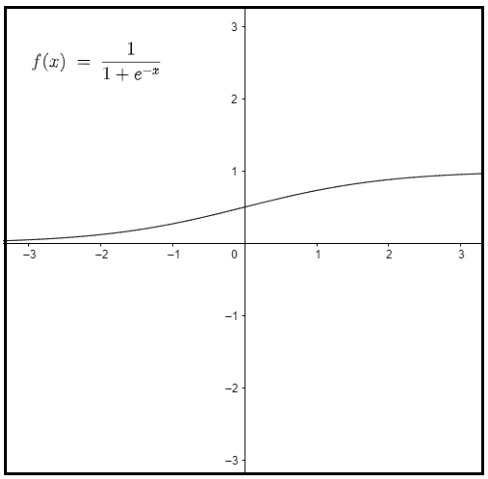


Figura A.5: Función Sigmoide

Función Tangente hiperbólica

Esta función de activación transforma el valor de pre-activacón en uno de salida entre el rango de -1 y 1. A diferencia de la función sigmoide el gradiente de la función tangente hiperbólica es más pronunciado y es más frecuente usar esta función debido a que sus gradientes no están restringidos a variar en una determinada dirección y está centrada en cero [22]. La ecuación que la define está dada por la ecuación A.3 [24] y su presentación se puede ver en la Figura A.6.

$$tanh(x) = \frac{sinh(x)}{cosh(x)} = \frac{e^{x} + e^{-x}}{e^{x} + e^{-x}} \tag{A.3}$$

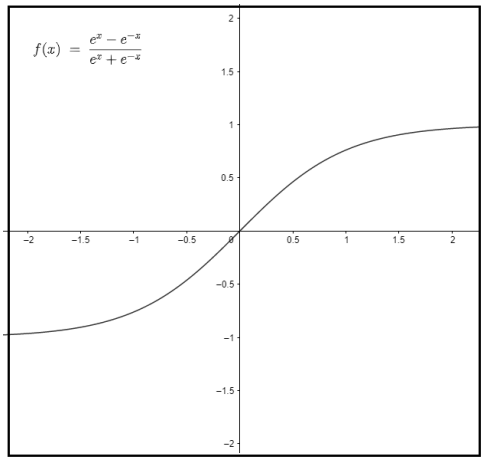


Figura A.6: Función Tangente Hiperbólica

Función ReLU

Por su nombre en inglés *Rectified Linear Unit* (ReLU), es una de las funciones que se utiliza con mayor frecuencia dentro de las capas ocultas de las redes neuronales, su valor es cero si los valores de entrada son negativos y con valores de entrada positivos permite su paso sin cambiarlos, es decir si el valor de entrada "x" es positivo devolverá el mismo valor de "x" [17].

En comparación con las demás funciones, ReLU suele ser más eficiente debido a que no todas las neuronas se activan al mismo tiempo. En algunos casos, el valor del gradiente es cero, por lo que los pesos y los sesgos no se actualizan durante el paso de retro propagación en el entrenamiento de redes neuronales [22]. La función ReLU está dada por la ecuación A.4 y su comportamiento se puede ver en la Figura A.7 .

$$f(x) = max(0, x) \tag{A.4}$$

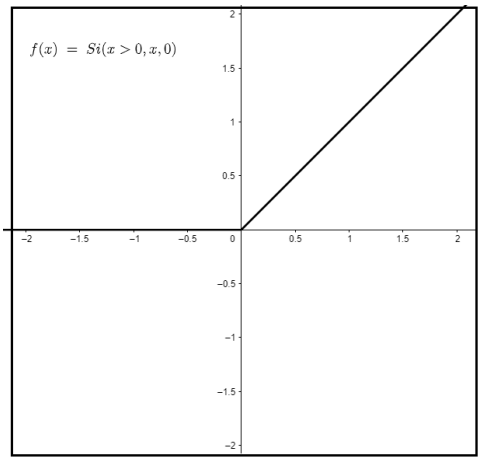


Figura A.7: Función ReLU

Función Leaky ReLu

Esta función es una versión mejorada de la función ReLU, la diferencia es que en la versión clásica de ReLU los valores negativos son puestos en 0, mientras que para la versión Leaky se define una pequeña pendiente para estos valores [22]. Se expresa matemáticamente por las ecuaciones en A.5 y su comportamiento se muestra en la figura A.8.

$$\begin{aligned} f(x) &= x, x \geq 0 \\ f(x) &= 0.01x < 0 \end{aligned} \tag{A.5}$$

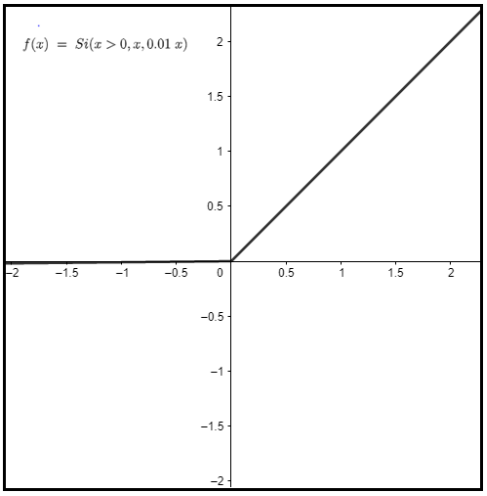


Figura A.8: Función Leaky ReLU

Función Softmax

Esta función de activación es utilizada en las capas de salida para problemas de clasificación multiclase, por lo tanto, esta topología tendrá en la última capa el mismo número de neuronas que de clases en las que se quieren clasificar. Esta función nos devuelve la probabilidad (en un rango de 0 y 1) de que el valor de entrada pertenezca a una de las clases, sabiendo que la suma de estas probabilidades debe ser igual a 1 [21]. Está dada por la ecuación A.6.

$$\sigma(Z)j = \frac{e^{Zj}}{\sum_{n=1}^{N} e^{Zk}}, j = 1..., N \tag{A.6}$$

A.3.2. Función de pérdida

Se ha mencionado hasta ahora el proceso para obtener la salida de una red neuronal, el significado que se le otorga a esta salida es representado de acuerdo al conjunto de entrada con el que se trabaje, esta salida estimada se denota como $\hat{y}$. Se dice que si una red neuronal está calibrada correctamente el valor de $\hat{y}$ será igual al resultado esperado por la red el cual se denota como, y suponga que se tiene una red con una única salida, la cual determinará si la entrada otorgada a la red es clasificada como un rostro, $y = 1$ indica que el objeto presentado es un rostro, en caso contrario $y = 0$. Por lo que si a la red le otorgamos un rostro el valor esperado de $\hat{y} = 1$, no obstante, en la práctica es difícil que $\hat{y} = y$, por lo cual existe la función de perdida cuyo objetivo es calcular el error que existe entre estos valores. Esta función de perdida también es conocida como función de costo, a continuación se explicarán dos de ellas, costo cuadrático y entropía cruzada [17].

A.3.3. Costo cuadrático

Es una de las funciones de costo más sencillas de calcular, conocida también como error cuadrático medio. Está representado por la Ecuación A.7.

$$C = \frac{1}{n}\sum_{i=1}^{n}(yi - \hat{yi})^2 \tag{A.7}$$

Para cada instancia i dada, se calcula el error entre el valor real y_i y la salida que estima la red $\hat{y_i}$, esta diferencia se elevará al cuadrado por dos principales razones, en primer lugar, sirve para asegurar que el valor obtenido será positivo independientemente de si y o $\hat{y}$ es mayor y en segundo lugar debido a que el cuadrado penaliza las grandes

diferencias entre estos valores mucho más severamente que las pequeñas diferencias.

Una vez obtenido el error cuadrático para cada instancia i se calcula el costo medio C en todas las n instancias realizando la suma de todas las instancias y dividiéndolas entre número total de instancias que se tienen [17].

A.3.4. Entropía cruzada

Es una opción de función de costo más popular. Esta dada por la Ecuación A.8.

$$C = -\frac{1}{n}\sum_{i=1}^{n}\left[yi\ln\hat{yi} + (1-yi)\ln 1 - \hat{yi}\right] \tag{A.8}$$

Al igual que el costo cuadrático, la divergencia de $\hat{y}$ de y corresponde a un mayor costo, además de manera similar al uso del cuadrado en el costo cuadrático, el uso del logaritmo natural ln en el costo de la entropía cruzada hace que las diferencias más grandes entre $\hat{y}$ y y se asocien con un costo exponencialmente mayor. El costo de la entropía cruzada está estructurado de modo que cuanto mayor sea la diferencia entre $\hat{y}$ y y, más rápido podrá aprender la neurona.

Para recordar más facilmente, cuanto mayor es el costo, más rápidamente aprende una red neuronal que incorpora el costo de la entropía cruzada [17].

A.3.5. Descenso de gradiente

El descenso del gradiente es un algoritmo de optimización, cuyo objetivo es encontrar el mínimo de la función de costo ajustando los parámetros del modelo, es conocido como la forma más común de optimizar a las redes neuronales, demostrando que las soluciones obtenidas por este algoritmo son óptimas [18] [25]. Este algoritmo requiere que la funciones sean diferenciables ya que el proceso se basa en las derivadas de la función, a lo que se le conoce como gradiente, este dará la pendiente de la función en un punto, o la dirección a la cual moverse, generalmente esta dirección será para llegar al punto máximo, por lo que si se desea encontrar el mínimo de la función se seguirá al negativo del gradiente o el punto donde su derivada sea 0 [20]. El comportamiento de este algoritmo puede verse en la figura A.9.

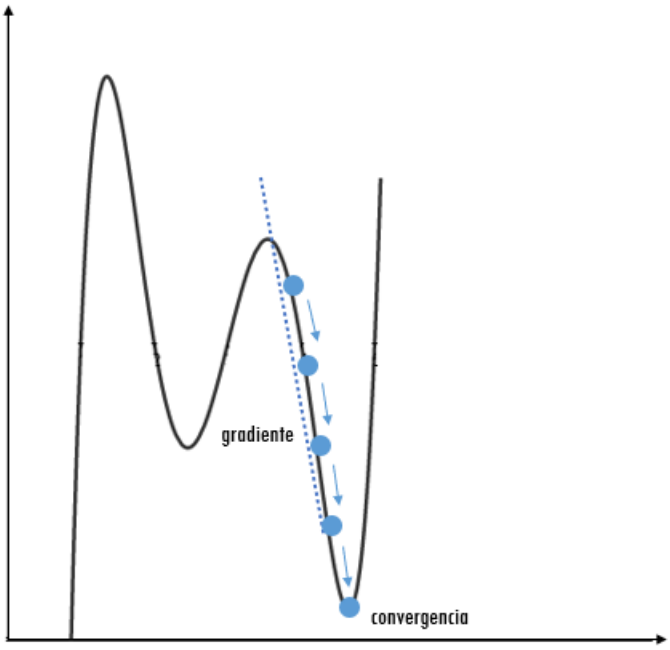


Figura A.9: Descenso del gradiente

Un hiperparámetro considerado dentro del descenso del gradiente es la taza de aprendizaje, esta determina el tamaño del paso, es decir la distancia a la que se va a realizar el desplazamiento [15], pero escoger el valor adecuado para este hiperparámetro no es una tarea sencilla, cuando la taza de aprendizaje suele tener un valor muy pequeño al algoritmo le tomará muchas iteraciones converger, además de que si la superficie de respuesta tiene muchos mínimos locales tiene mayor probabilidad de quedarse atrapado en alguno de ellos. Por otro lado, si se establece una taza de aprendizaje muy grande se darán pasos o saltos de gran tamaño, que lo harán llegar a otro extremo con la posibilidad de encontrarse con valores más grandes que los anteriores, es decir se alejará cada vez más del valor mínimo y el algoritmo divergirá sin encontrar una solución óptima [18]. Es por eso que se debe de escoger una taza de aprendizaje apropiada para evitar encontrarnos con estas situaciones, un valor comúnmente utilizado para la taza de aprendizaje es de 0.01. En las redes neuronales, la taza de aprendizaje servirá para modificar los valores de los sesgos y los pesos, la actulización se hará con la ecuación A.9 donde W es denominado como el peso [23].

$$Wnuevo = Wviejo - taza\ de\ aprendizaje * gradiente \tag{A.9}$$

El algoritmo del descenso del gradiente tiene algunas variantes, las cuales se diferencían

dependiendo del número de datos que son utilizados para calcularlo.

Descenso del gradiente por lotes (o batch) por su nombre en inglés *Batch Gradient Descent* también conocido como *Vanilla Gradient Descent* utiliza todo el conjunto de datos para calcular el gradiente y lo hace en solo una actualización para después seguir la pendiente y tomar el camino con mayor inclinación. Esta variante funciona muy bien para una superficie de error cuadrática simple (funciones convexas) y es sensible a los puntos de silla en las funciones no convexas, si se cae en alguno de ellos esto puede provocar que se encuentre una solución prematura, el descenso puede ser muy lento e intratable para conjuntos de datos que no caben en la memoria [15] [25].

Descenso del gradiente estocástico o *SGD (Stochastic Gradient Descent)* a diferencia de la variante anterior, si se tiene un conjunto de datos grande esta solamente toma una muestra aleatoria de los datos de entrenamiento y calcula el gradiente sobre ese conjunto por lo que hace que el algoritmo sea más rápido, pero debido a que es estocástico, es menos regular que el descenso por lotes. SGD le permite moverse a mínimos locales nuevos y mejores, pero nunca se asienta por lo que hace que no pueda converger sino simplemente terminar muy cerca del mínimo, por lo tanto, cuando el algoritmo pare, la solución será buena pero no óptima. Se menciona que se ha demostrado que cuando se disminuye la taza de aprendizaje en SGD, muestra el mismo comportamiento de convergencia que el descenso de gradiente por lotes convergiendo a un mínimo global o local para funciones convexas y no convexas [25] [18].

Descenso del gradiente por mini lotes (o mini-batch) toma ciertas características de las variantes pasadas, en ésta se toman pequeños conjuntos aleatorios llamados mini lotes para calcular los gradientes, encontrará una mejor solución, más cercana al mínimo, a diferencia de SGD, pero puede ser más difícil para salir de los mínimos locales. Al igual que SGD el descenso de gradiente por mini lotes no se asienta, es decir, continúa moviéndose y debe escogerse una buena taza de aprendizaje para que pueda alcanzar una solución mínima buena, recordando que gradiente por lotes tardará más tiempo en cada paso [18].

El comportamiento de estas variantes se puede ver en la Figura A.10.

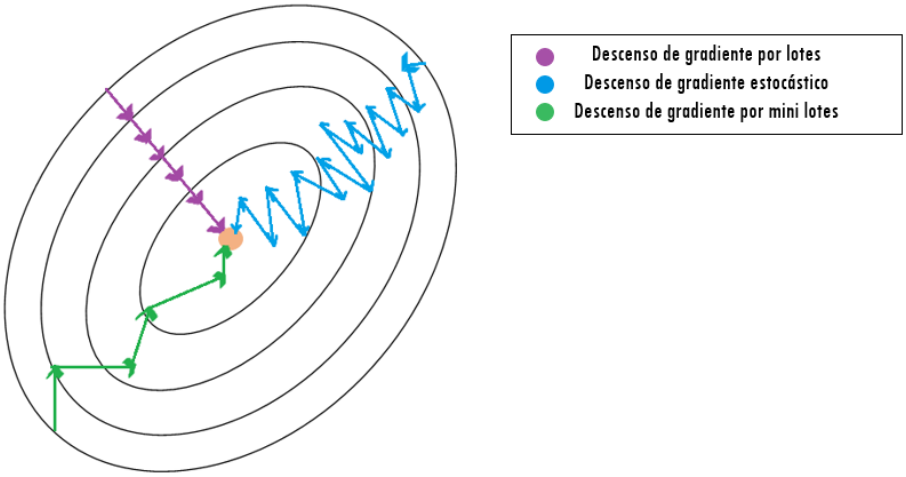


Figura A.10: Descenso del gradiente - variantes.

A.3.6. Backpropagation

El algoritmo de retropropagación o *backpropagation*, es la base para el aprendizaje de las redes multicapa, fue descubierto por primera vez por Paul Werbos en su tesis doctoral en 1974 pero no tuvo el reconocimiento que merecía y fue hasta 1986 que David Rumelhart, Geoffrey Hinton y Ronald Williams publicaron el algoritmo que se hizo popular, y se sigue utilizando hoy en día para realizar el entrenamiento de las redes neuronales [21] [18]. Este algoritmo consiste en tomar el valor de pérdida final y trabajarlo hacía atrás, desde la última capa hacia las capas anteriores aplicando la regla de la cadena con el objetivo de reducir los costos y ajustar los parámetros de las neuronas en toda la red [20] [17].

El algoritmo emplea un ciclo de propagación el cual se divide en dos fases: la fase de propagación hacia adelante y la de propagación hacía atrás:

La fase de propagación hacia adelante consiste en tomar un conjunto de datos de entrada, asociarlos con sus pesos y sesgos para poder realizar el cálculo de la suma ponderada junto con la función de activación y obtener las salidas de la neurona en esta capa, después los resultados pasarán a la capa siguiente donde de igual forma se calculará su salida y se propagará hacia la siguiente capa, y así sucesivamente hasta obtener la salida de la última capa, la capa de salida [18].

Para la fase de propagación hacía atrás, el algoritmo mide el error de salida de la red (es decir, utiliza una función de pérdida que compara la salida deseada y la salida real de la red, y devuelve alguna medida del error), después calcula cuánto contribuyó

cada conexión de salida al error analíticamente aplicando la regla de la cadena. Luego, el algoritmo mide cuánto de estas contribuciones de error provienen de cada conexión en la capa de abajo, nuevamente usando la regla de la cadena, trabajando hacia atrás hasta que el algoritmo llega a la capa de entrada. Este paso inverso mide eficientemente el gradiente de error a través de todos los pesos de conexión en la red propagando el gradiente de error hacia atrás a través de la red. Finalmente, el algoritmo realiza un paso de descenso de gradiente para ajustar todos los pesos en la red, utilizando los gradientes de error que acaba de calcular. [18].

Apéndice B

Redes Neuronales Convolucionales

Las redes neuronales convolucionales conocidas también como *Convolutional Neuronal Networks* (CNN) fueron creadas en 1998 por Yann Lecun y otros en base a las ideas de David H. Hubel y Torsten Weisel, presentadas en 1968 en donde ellos exploraron el funcionamiento de la corteza visual del gato y encontraron conexiones entre las actividades en un área pequeña del cerebro y las actividades en pequeñas regiones del campo visual [21]. La primera arquitectura basada en esta inspiración biológica fue el neocognitrón que se generalizó en la arquitectura LeNet-5. Las redes neuronales convolucionales han sido las más exitosas de todos los tipos de redes neuronales y sus usos principales se emplean para la resolución de tareas de reconocimiento de patrones basadas en imágenes, detección de objetos, clasificación de imágenes incluso en procesamiento de texto.

B.1. Arquitectura

Este tipo de redes neuronales tienen un comportamiento parecido al de las redes tradicionales *feed-forward*. Se encuentra organizada por 3 tipos de capas: la capa de convolución, la capa de agrupamiento *(pooling)* y la capa ReLU [19].

El valor de entrada estará compuesto por una imagen, la cual se puede representar como una matriz de pixeles, en donde el valor de cada pixel va de 0 a 255 que representa la intensidad del color, ese valor se normalizará para poder trabajar con la red neuronal. La configuración clásica para las capas convolucionales está presentada por una red de 3 dimensiones que contiene altura, anchura y profundidad, si se trabaja con imágenes a color. La profundidad se refiere al número de canales en cada capa, como el número

de colores primarios: rojo, verde y azul (RGB) en la imagen de entrada o el número de mapas de características en las capas ocultas [19], aunque también se encuentran configuraciones 2D cuando la imagen de entrada se trata de una imagen a blanco y negro, las cuales son denominadas capas convolucionales 2D o capas convolucionales planas y las 4D hiperespacial [21].

B.2. Convoluciones

Como se mencionó anteriormente, la red recibe una imagen de pixeles en la capa de entrada, estos pixeles no están procesados. Si se deseara hacer la conexión entre la capa de entrada y la capa oculta se tendrían demasiados parámetros, por lo que se buscó un método más eficiente en donde los parámetros son organizados en conjuntos de unidades estructurales tridimensionales conocidas como filtros o kernels, éstos son cuadrados y típicamente el tamaño es más pequeño que la capa a la que se le está aplicando, pero la profundidad de este seguirá siendo la misma. La operación de convolución consiste en tomar grupos de pixeles de la imagen de entrada (o de las capas ocultas) moviendo de izquierda a derecha y una vez terminada la fila debe moverse hacia abajo de igual forma e ir realizando un producto escalar, este proceso está ilustrado en la Figura B.1. Se obtendrá como resultado un conjunto de matrices el cual es conocido como mapa de características, es decir, las convoluciones permiten detectar y reconocer características independientemente de la posición en la imagen. Todo este procedimiento pasa dentro de la capa convolucional [21].

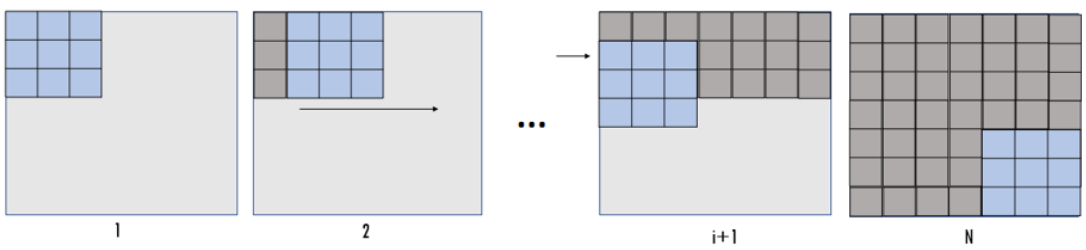


Figura B.1: Capa de convolución

B.3. Capa ReLU

El entrenamiento de la capa significa entrenar los campos receptivos locales de la capas con sus pesos y sesgos completamente conectados utilizando la función de activación ReLU *(Rectified Linear Unit)*, la activación no es muy diferente de cómo se aplica en una red neuronal. La función ReLU de x es simplemente el valor máximo de 0 y x, en donde significa que regresa 0 si la entrada es negativa o de lo contrario se regresará la entrada sin procesar. Se aplicará la fórmula B.1 [21]:

$$p(X) = max(x, 0) \tag{B.1}$$

B.4. Pooling

El objetivo de la capa de agrupación es reducir la resolución de las características del dato de entrada y de esa manera se reduce el número de parámetros. Existen dos maneras de trabajar con esta capa: *max pooling and average pooling* [21]. La diferencia entre ellas es cómo se realiza el cálculo de agrupamiento, para este trabajo hablaremos de *max pooling* porque es técnica más empleada.

B.5. Max pooling

La capa de *max pooling* se maneja comúnmente como una matriz de 2x2, no obstante, los valores de esta matriz son definidos al gusto del usuario, pero tomando en cuenta que su dimensionalidad debe ser menor que la original, para ejemplificar se tomará una matriz de 2x2 y se tomarán grupos de cuatro pixeles como se puede ver en la Figura B.2. En segundo lugar se hará un recorrido de igual forma que el realizado en la parte de las convoluciones, es decir, de izquierda a derecha y al terminar la fila continúa con el primer elemento de la siguiente fila, para el proceso de *max pooling* se tomará el valor más alto de los cuatro pixeles, estas capas generan una imagen de la mitad del tamaño que la original sin afectar al número de canales, una pequeña representación de este método está en la Figura B.2. La diferencia entre este método y el *average pooling* es que en vez de tomar el valor más alto se calculará el valor promedio de los elementos.

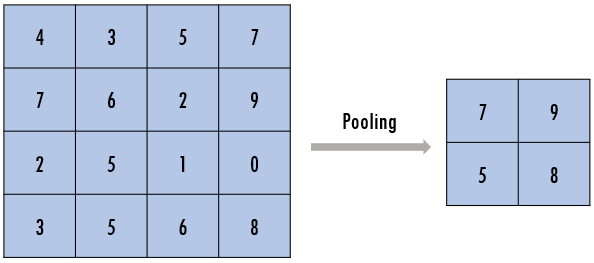


Figura B.2: *Max pooling*

Por lo general una red neuronal convolucional se compone de una capa convolucional seguida de una de agrupación, es decir, se vuelven a realizar más convoluciones y así después de varias capas se obtendrá una imagen más pequeña y conforme pasa esto los mapas de características podrán reconocer patrones y formas más elaboradas.

B.5.1. Capa totalmente conectada

Finalmente se tomará la última capa, esta capa dejará de ser tridimensional y pasará a ser una capa de neuronas tradicional, funcionará de la misma manera que una red *feed-forward*, las neuronas de estas capas estarán conectadas con sus respectivos pesos. En esta capa se puede hacer uso de las funciones de activación logística, softmax o lineal, dependiendo de la naturaleza de la aplicación ya sea clasificación o regresión. La Figura B.3 muestra la arquitectura de una red convolucional.

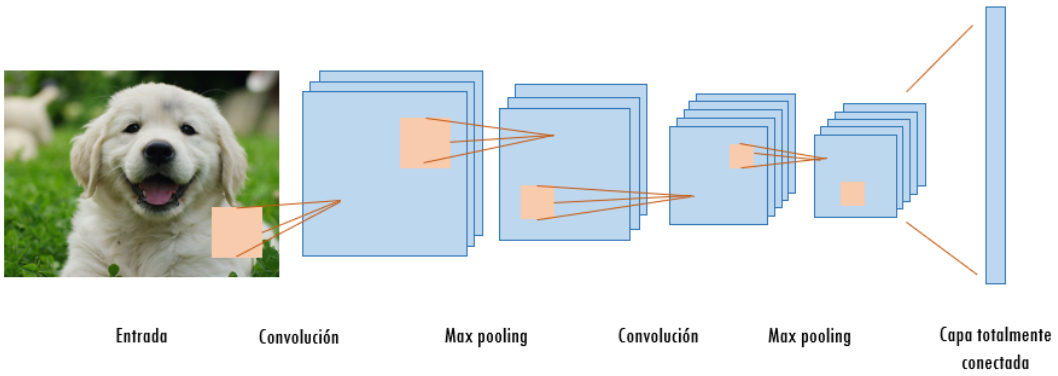


Figura B.3: Red neuronal convolucional.

Apéndice C

Configuraciones del Modelo SAEHD

En este capítulo se mencionan las configuraciones del Modelo SAEHD en el orden que aparecen al solicitar entrenar con este modelo. Algunas de estas configuraciones dependen de la opción anterior, es decir, dependiendo de como se configure aparecerán o no [X].

- **Copia de seguridad automática cada N horas (0-24)**. Permite habilitar copias de seguridad del modelo, y realizar una copia de respaldo en caso de que se necesite disponer de la información si llega a haber una pérdida. El valor predeterminado de esta opción es 0 el cual significa deshabilitado, de otro modo deberá seleccionar el valor de N.

- **Escribir historial de vista previa (y/n):** Guarda las imágenes de vista previa durante el entrenamiento cada pocos minutos. Solo se puede seleccionar y – sí o n – no, en caso de seleccionar la opción no el modelo mostrará las caras en la vista previa de forma aleatoria, de otra forma se abrirá una ventana después de que se carguen los conjuntos de datos en donde se podrá elegir de forma manual.

- **Iteración de destino:** Esta opción permite establecer cierta cantidad de iteraciones para que el modelo entrene, el valor predeterminado es 0 y al aceptar dejar ese valor el entrenamiento se ejecutará hasta que el usuario lo detenga manualmente.

- **Voltear caras fuente aleatoriamente (y/n):** Voltea aleatoriamente las caras fuente de forma horizontal, esta opción ayuda a cubrir los ángulos presentes en el conjunto de datos destino, lo cual resulta útil especialmente si el conjunto de datos no tiene condiciones de iluminación diferente, por otra parte puede resultar

contraproducente debido a que puede hacer que los resultados no parezcan naturales, ya que las caras nunca son perfectamente simétricas. Se recomienda no utilizarlo si el conjunto de caras es diverso, por lo que el valor por defecto es no.

- **Voltear caras destino aleatoriamente (y/n):** Voltea aleatoriamente las caras destino de forma horizontal lo cual puede mejorar la generación cuando la opción anterior esta deshabilitada, es decir si la opción de voltear caras fuente aleatoriamente está deshabilitada, esta opción estará habilitada por defecto.

- **Tamaño del lote:** El tamaño del lote afecta sobre la cantidad de caras que se comparan entre sí en cada iteración. El valor más bajo que se puede seleccionar en esta configuración es de 2 y puede ir tan alto como lo permita la GPU. Entre mayor sea el tamaño del lote la calidad será mejor, pero el entrenamiento será más lento ya que requiere mayor tiempo por iteración. Se recomienda utilizar un valor no menor a 4, entre los valores más adecuados están entre 6 y 12 con un valor predeterminado de 8.

- **Resolución (64-640):** Esta opción es ajustable durante el entrenamiento, afecta la resolución de las caras intercambiadas. Cuanto mayor sea la resolución más detallada será la cara aprendida, pero de igual forma será mucho más pesado y tardado. La resolución puede aumentar de 64x64 a 640x640 en incrementos de 16 para variantes de arquitecturas regulares y de tipo -U y de 32 para variantes de arquitecturas de tipo – D y -UD.

- **(Tipo de cara /h / mf/ f/ wf/ head):** Existen 5 opciones para establecer el área de la cara que se desea entrenar: media cara, mitad media cara, cara completa, toda la cara y cabeza.

    1. **Media cara (h):** Se entrena desde la boca hasta las cejas, en algunos casos puede cortar la parte superior o inferior de la cara (cejas, barbilla o la parte inferior de la boca)

    2. **Mitad media cara (mf):** Cubre una porción de la cara 30 % más grande que la media cara. Esta opción hace que no ocurran algunos de los cortes sin embargo todavía cabe la posibilidad de que los haya.

3. **Cara completa (f):** Cubre la mayor parte de la cara, excluyendo la frente y en algunos casos parte de la barbilla, aunque en comparación con los anteriores sucede escasas ocasiones, regularmente se presenta cuando el personaje hace una abertura muy grande con la boca. Esta opción se recomienda cuando algunos de los datos fuente o destino tiene cabello cubriendo la frente.
4. **Toda la cara (wf):** El área de cobertura es mayor, abarca toda la cara incluida la frente y toda la cara desde el costado hasta la parte de las orejas.
5. **Cabeza (head):** Se utiliza para realizar un intercambio de toda la cabeza, no es recomendado para personas con cabello largo. No cambia de forma dependiendo del ángulo.

- **Arquitectura AE (df / liae / df-u / liae-u / df-d / liae-d / df – ud / liae – ud):** Esta opción muestra las dos arquitecturas de aprendizaje principales: DF y LIAE con sus variantes -U, -D y -UD.

  1. **DF:** Este modelo proporciona un intercambio de caras, no hace la transformación de éstas, por lo mismo requiere que las caras utilizadas de destino y fuente tengan una forma similar, y los rasgos de boca, nariz, ojos pueden no ser tan similares. Tiene un buen funcionamiento en tomas frontales y requiere que el conjunto de datos que se utilice sea amplio o tenga todos los ángulos que requiere el video fuente para obtener buenos resultados. Aunque la arquitectura LIAE obtiene mejores resultados laterales.
  2. **LIAE:** Este modelo es más flexible que DF, es decir, no se pide que la forma de la cara fuente sea similar a la destino. Sin embargo, es deseable que la forma de los rasgos faciales si sean similar para obtener mejores resultados. A pesar de que el parecido con las caras fuentes suele ser menor que el modelo anterior esta arquitectura maneja mejor los perfiles laterales.
  3. **DF-U / LIAE- U:** El objetivo de esta variante es mejorar la similitud con las caras de origen.
  4. **DF-D / LIAE-D:** El objetivo es mejorar el rendimiento al duplicar la resolución sin costo de cálculo adicional y tener un rendimiento similar. El entrenamiento de esta variante es más tardado ya que el modelo debe entrenarse con anticipación.

5. **DF-UD / LIAE-UD:** Es una combinación de las variantes anteriores para obtener la mayor semejanza y una mayor resolución y rendimiento. También requiere un entrenamiento previo.

Las 4 configuraciones controlan las dimensiones de la red neuronal y afectan la capacidad de aprendizaje por lo que modificarlas puede hacer que se tenga un gran impacto en el rendimiento y la calidad, sin embargo, para saber o medir el efecto de su influencia se necesitan realizar pruebas exhaustivas.

1. **Dimensiones del codificador automático (32-1024):** El ajuste del codificador automático afecta en la capacidad general del modelo para aprender los rostros. El valor predeterminado es de 256.
2. **Dimensiones del codificador (16-256):** Se puede ajustar con un valor de 16 a 256, como valor de defecto es seleccionado el valor de 64. Cambiar este valor afecta la capacidad del modelo para aprender la estructura general de los rostros.
3. **Dimensiones del decodificador (16-256):** Los valores con los que se puede ajustar no varían entre la opción anterior, este valor afecta la capacidad del modelo para aprender mayor detalles. El valor por defecto también es 64.
4. **Dimensiones de la máscara del decodificador (16-256):** El valor predeterminado por este ajuste es de 22. Esta configuración altera la calidad de las máscaras aprendidas, en ocasiones afecta algunos otros aspectos del entrenamiento.

- **Prioridad de ojos y boca (y/n):** Entrena ojos, boca y dientes con una prioridad más alta. Puede mejorar la calidad y nitidez del rostro.
- **Distribución uniforme de guiñadas de muestra (y/n):** Ayuda con el entrenamiento de caras de perfil y fuerza al modelo a entrenar uniformemente en todas las caras dependiendo de sus guiños y prioriza las caras de perfil, las caras frontales pueden entrenar con mayor tiempo lo cual es un efecto que se puede habilitar durante el preentrenamiento. El valor predeterminado es no.
- **Colocar los modelos y optimizadores en el GPU (y/n):** Habilita el optimizador y pone toda la carga en su GPU, lo que mejora el rendimiento, es decir, el

tiempo en el que se realizarán las iteraciones, aunque esto hará que el consumo de VRAM sea mayor. La opción predeterminada es si, pero si se decide deshabilitar esta opción parte del trabajo que realice el optimizador se descargara en la CPU, lo que hará que logre un mayor tamaño de lote o ejecute modelos más exigentes a costa de tiempos más largos. De igual forma si esta opción es habilitada y ocurre el error de memoria insuficiente, se deberá deshabilitar la configuración si no desea reducir el tamaño del lote, así podrá ejecutarse el modelo sin errores a una velocidad menor.

- **¿Usar el optimizador AdaBelief? (y/n):** AdaBelief aumenta la precisión del modelo y calidad de los rostros entrenados, esta opción está habilitada por defecto y reemplaza el optimizador RMSProp predeterminado. Hacer uso de este optimizador aumenta el uso de la VRAM, lo que requiere que los modelos nuevos sean entrenados con un tamaño de lote más bajo para evitar errores de memoria insuficiente. Siempre se debe usar esta opción desde el comienzo y no debe de deshabilitarse durante el entramiento. La siguiente configuración (uso de la tasa de aprendizaje de abandono) se recomienda mantener deshabilitada.

- **¿Usar la tasa de aprendizaje de abandono? (y/n):** El valor por defecto de esta opción es no. Se utiliza para acelerar el entrenamiento de rostros y reduce el movimiento temblores en las caras, así como el parpadeo de la iluminación. Se utiliza en 3 casos:

  1. Antes de deshabilitar RW, cuando los valores de perdida ya no mejoran mucho, esto puede ayudar al modelo a generalizar un poco más los rostros.
  2. Después de deshabilitar RW y se haya entrenado el modelo lo suficientemente bien, habilitarlo cuando el entrenamiento va a llegar a su fin, da como resultado rostros más detallados y estables.
  3. Después que se haya entrenado un poco con la tasa de aprendizaje de abandono y las caras se vean tan bien como sea posible, GAN y la tasa de aprendizaje deben mantenerse habilitados mientras se ejecuta.

  Esta opción afecta el uso de VRAM por lo que si se encuentra algún error puede ser ejecutada en la CPU, recordando que el tiempo de iteración se reducirá.

- **Habilitar la deformación aleatoria de muestras (y/n):** La deformación aleatoria se utiliza para generalizar un modelo y este aprenda correctamente las características y expresiones faciales en la etapa de entrenamiento inicial. Está habilitado como configuración inicial, pero en este caso puede presentar problemas para aprender detalles finos. Se recomienda tenerlo habilitado el tiempo necesario a medida que sus rostros siguen mejorando, para ello se debe observar la disminución de los valores de perdida y los rostros en la ventana de vista previa. Una vez que se vean los rostros correctos y que el valor de perdida no disminuye, esta opción puede ser deshabilitada de lo contrario el modelo no aprenderá los detalles. En dado caso que arruine los resultados se puede volver a habilitar cuando desee reutilizar ese modelo para entrenar un nuevo video de destino con la misma fuente o bien cuando ambos rostros destino y fuente sean nuevos.

- **Potencia de GAN (0.0 -10.0):** Esta opción es una forma adicional para obtener caras más detalladas o nítidas. Es ajustable en una escala de 0.0 a 10.0 y solo debe habilitarse una vez que el modelo este medio entrenado. La configuración deformación aleatoria de muestras debe ser deshabilitada y se deberá habilitar la tasa de aprendizaje por abandono. Se recomienda utilizar valores bajos como 0.01 así como haber realizado una copia de seguridad antes de entrenar.

- **Potencia de estilo de cara y potencia de estilo de fondo (0.0 – 100.0):** Controla la transferencia de estilo de la cara o la parte de fondo de la imagen. Se utiliza para transferir el estilo de sus caras destino a la cara aprendida final lo que puede mejorar la calidad y el aspecto del resultado después de la fusión. Transferirá cierta información de color e iluminación de destino a la cara del resultado, lo que ayuda con la coincidencia de colores y a reducir el parpadeo. Se recomienda no usar valores superiores a 10, empezando por valores pequeños de 0.001 y aumentar progresivamente. Esta función tiene un impacto en el rendimiento y su uso aumentará el tiempo de iteración y puede requerir que reduzca el tamaño de su lote o deshabilite el optimizador de GPU.

- **Transferencia de color para el conjunto de caras fuente (none/ rct/ lct/ mkl/ idt/ sot):** Esta función se utiliza para hacer coincidir los colores de los rostros fuente con los rostros destino para que el resultado tenga un tono de piel similar

al destino y estabilizar el color cuando la cara se mueva en distintos ángulos, esto debido a si el conjunto de datos fuente contienen diferentes condiciones de luz o de clasificación de manera diferente en color.

- **Habilitar el recorte de degradado (y/n):** El objetivo de esta función es evitar el colapso del modelo que pueda ocurrir. No tiene gran impacto en el rendimiento por lo que si no desea usarlo puede habilitar las copias de seguridad automáticas ya que si el modelo colapsa la información del entrenamiento se pierde y deberá comenzar de nuevo. El colapso puede ocurrir cuando se utilizan configuración de potencia de estilos, por lo que si está usando se recomienda habilitarlo.

- **Habilitar el modo de preentrenamiento (y/n):** Permite el proceso de preentrenamiento que usa un conjunto de datos de caras de personas aleatorias para entrenar inicialmente su modelo. Después de hacer 200 mil iteraciones, este modelo se puede usar al comenzar a entrenar con los datos fuente y destino reales. Ahorra tiempo porque no es necesario que empiece a entrenar de 0 cada vez, sino que el modelo sabrá como deben verse las caras y por lo tanto acelerará la etapa de entrenamiento inicial. Aun así, la configuración inicial ese encuentra deshabilitada.

Apéndice D

Encuesta

La tecnología DEEPFAKE permite generar manipulaciones en videos, lo cual además traer beneficios en la parte de entretenimiento, puede generar desinformación. En este trabajo de investigación se realizó una encuesta con el objetivo de analizar cuantas personas podrían ser capaz de detectar un *deepfake*. La encuesta[1] esta conformada por 9 preguntas y fue dividida en dos partes. La parte 1, incluye cuatro preguntas, en la que se presentan dos videos, uno original y un *deepfake* generado a partir de éste. El encuestado debe indicar cuál de los dos videos es el falso. La parte 2, incluye 5 preguntas con un solo video, en esta parte el encuestado debe indicar si el video que se le presenta es original o si se trata de un *deepfake*.

Los participantes de esta encuesta fueron separados en dos grupos. Los integrantes del grupo B, estudiantes de maestría y doctorado, los cuales al contestar la encuesta tenían conocimiento sobre los *deepfakes*, ya que se les hicieron 3 presentaciones explicando varios aspectos relacionados con éstos, como sus procesos de generación y los errores más comunes al producirlos. A diferencia de los del grupo A, que fue un público general, los cuales realizaron la encuesta sin recibir información adicional sobre qué son los *deepfakes*, cómo se generan o sus dificultades de procesamiento. Debido a las dificultades para coordinar los espacios y horarios para realizar las presentaciones, en el grupo B sólo se incluyeron a 20 personas, mientras que el grupo A requería de una participación breve, por lo cual, se pudo contar con la participación de 105 personas.

La Figura D.1 presenta el porcentaje de respuestas correctas para cada pregunta y por cada grupo. Se puede observar que el % de asertividad del grupo B fue mayor al del grupo

[1] https://forms.gle/wfSvb8aK9QuBLPCR9

A en la mayoría de las preguntas. Por otro lado, hay preguntas que parecen ser más difíciles que el resto, como fue el caso de la pregunta 7, la cual, tuvo una asertividad por debajo del 40 % por ambos grupos, mientras que la pregunta 4 fue respondida correctamente por más del 80 % de los encuestados.

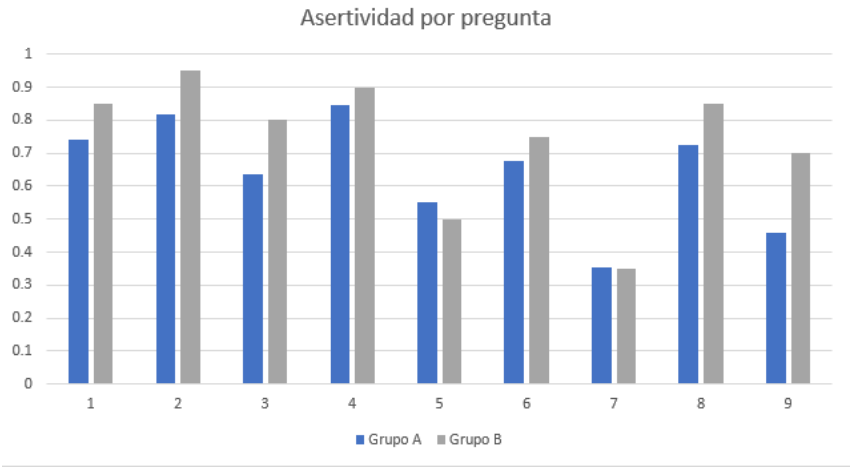
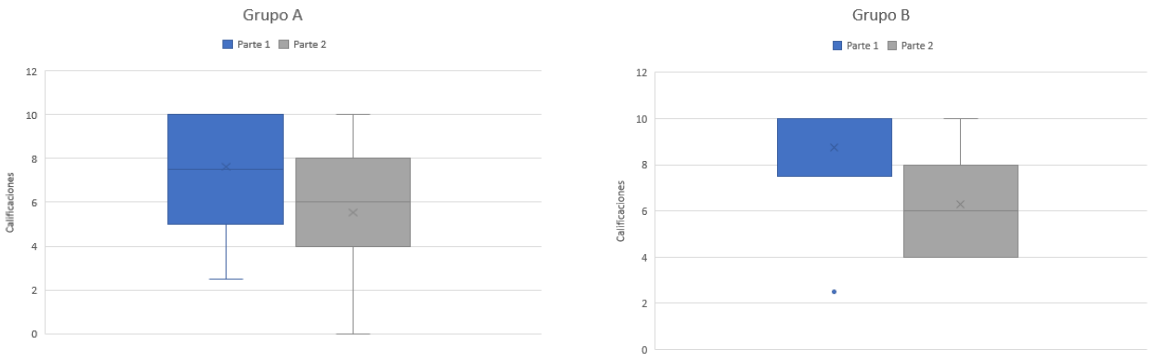


Figura D.1: Histograma de asertividad.

Además, se percató que los participantes tuvieron mayor facilidad de reconocer el video manipulado en la parte 1, lo cual implica que ante la presencia del video original, es más sencillo identificar una falsificación. Para confirmar se analizaron las respuestas de la parte 1 y la parte 2 por separado y se le asignó a cada encuestado una calificación para cada bloque, dependiendo de su número de aciertos.

Se utilizó la prueba de *(*Wilcoxon) para comparar los rangos medios de los conjuntos A y B y así saber cual bloque de preguntas fue más sencillo de responder. Los resultados se ilustran con los diagramas de cajas de las Figuras D.2a y D.2b, para los grupos A y B respectivamente. Se puede ver que las mejores calificaciones se obtienen en la parte 1, sin importar el grupo de encuestados. Esta observación se vio confirmada con la prueba de hipótesis de *Wilcoxon*, sobre las calificaciones del bloque 1 comparadas con las del bloque 2, y obtener que la hipótesis nula debe rechazarse para los dos grupos, A y B. Con lo cual se concluye que las preguntas del bloque 1 son estadísticamente diferentes, más sencillas, que las del bloque 2.



(a) (b)

Figura D.2: Calificaciones por bloque

Por otra parte, se observa que en la mayoría de los casos, el grupo B tiene un mejor desempeño al identificar los videos. Para determinar si esta observación es válida, se asignó una calificación, de 0 a 10, a cada participante, dependiendo del número de respuestas correctas en su encuesta. En la Figura D.3 se muestran los diagramas de cajas asociados a las calificaciones obtenidas por ambos grupos.

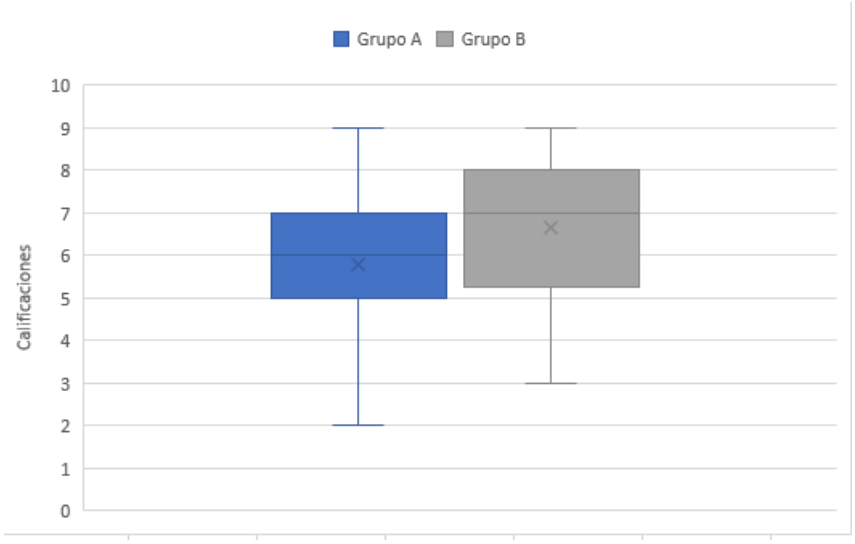


Figura D.3: Calificaciones por grupo.

Se puede observar un desempeño superior por parte del grupo B, ya que su mediana se encuentra claramente por arriba de la obtenida por el grupo A. Sin embargo, se determinó que era conveniente realizar una prueba de hipótesis para llegar a una conclusión definitiva. Ahora bien, debido a que el tamaño de las muestras en cada grupo es diferente, se consideró que lo mejor era realizar una prueba de *Mann-Whitney* para comprobar la igualdad de los

resultados entre los grupos A y B. En este caso, la hipótesis nula establece que la diferencia de las medianas no es estadísticamente diferente, pero fue rechazada al incorporar las calificaciones obtenidas por los grupos A y B. Por todo lo anterior, se puede concluir que el grupo B, el cual recibió información adicional sobre los deepfakes, obtuvo un mejor desempeño en su detección.

A los participantes del grupo A, se les hizo una pregunta adicional acerca de si conocían la tecnología *DEEPFAKE*. La Figura D.4 muestra el % de conocimiento acerca de los *deepfakes*, en donde se puede ver que el 60 % de los encuestados no conocían esta tecnología. Para finalizar la encuesta a este grupo se les informó sobre los *deepfakes* mostrando un video que explica de forma resumida las bases y métodos para su creación y detección.

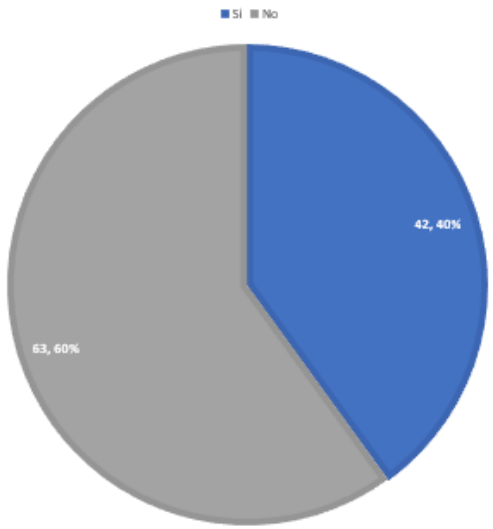


Figura D.4: % conocimiento sobre *deepfakes*.

Casa abierta al tiempo
UNIVERSIDAD AUTÓNOMA METROPOLITANA

ACTA DE EXAMEN DE GRADO

No. 00102
Matrícula: 2202800317

Estimación de caracteristicas óptimas para la generación de deepfakes.

En la Ciudad de México, se presentaron a las 11:00 horas del dia 13 del mes de diciembre del año 2022 en la Unidad Iztapalapa de la Universidad Autónoma Metropolitana, los suscritos miembros del jurado:

DRA. BIBIANA OBREGON QUINTANA
DR. ROMAN ANSELMO MORA GUTIERREZ
DR. PEDRO LARA VELAZQUEZ

Bajo la Presidencia de la primera y con carácter de Secretario el último, se reunieron para proceder al Examen de Grado cuya denominación aparece al margen, para la obtención del grado de:

MAESTRA EN CIENCIAS (CIENCIAS Y TECNOLOGIAS DE LA INFORMACION)

DE: PAOLA POLET BELTRAN GARCIA

y de acuerdo con el articulo 78 fracción III del Reglamento de Estudios Superiores de la Universidad Autónoma Metropolitana, los miembros del jurado resolvieron:

APROBAR

Acto continuo, la presidenta del jurado comunicó a la interesada el resultado de la evaluación y, en caso aprobatorio, le fue tomada la protesta.

PAOLA POLET BELTRAN GARCIA
ALUMNA

REVISÓ

MTRA. ROSALIA SERRANO DE LA PAZ
DIRECTORA DE SISTEMAS ESCOLARES

DIRECTOR DE LA DIVISIÓN DE CBI

Roman Linares Romero
DR. ROMAN LINARES ROMERO

PRESIDENTA

BObregón
DRA. BIBIANA OBREGON QUINTANA

VOCAL

DR. ROMAN ANSELMO MORA GUTIERREZ

SECRETARIO

DR. PEDRO LARA VELAZQUEZ