

**UNIVERSIDAD NACIONAL DE SAN ANTONIO ABAD DEL  
CUSCO**

**FACULTAD DE INGENIERÍA ELÉCTRICA, ELECTRÓNICA,  
INFORMÁTICA Y MECÁNICA**

**ESCUELA PROFESIONAL DE INGENIERÍA INFORMÁTICA Y DE  
SISTEMAS**



**TESIS DE INVESTIGACIÓN**

**DESARROLLO DE UNA ARQUITECTURA DE RED NEURONAL  
CONVOLUCIONAL COMO UN MODELO DEL PROCESO CEREBRAL  
HUMANO PARA LA CLASIFICACIÓN DE EXPRESIONES FACIALES**

**Para optar al título profesional de:**  
Ingeniero Informático y de Sistemas

**Presentado por:**  
Br. Darwin Ttito Concha  
Br. Paul Dany Flores Atauchi

**Asesor:**  
Prof. Msc. Lauro Enciso Rodas

**Co-Asesor:**  
Prof. M.Eng E. Gladys Cutipa Arapa

**FINANCIADO POR EL CONSEJO DE INVESTIGACIÓN DE LA UNSAAC**

**CUSCO - PERÚ**  
2017

# Dedicatoria

*Dedico esta tesis a mi mamá Luz Marina, a mi tía Felicitas Magdalena y a mis hermanos Sharmely y Russel por el gran apoyo y motivación que siempre me brindan.*

*Paul Dany Flores Atauchi*

*Este trabajo está dedicado a mis padres Marina y Donato, mis hermanos Edison y Sayda. A todos ellos porque día a día me aconsejan y ayudan a ser una mejor persona.*

*Darwin Ttito Concha*

# Resumen

Las expresiones faciales son un medio de comunicación no verbal que muestran las emociones de una persona, estas expresiones ayudan a transmitir información en las interacciones inter personales y facilitan el entendimiento del significado del lenguaje hablado. Por lo que se considera que poder clasificar la expresión de un rostro sería una gran fuente de información para una posterior utilización. El objetivo del presente proyecto es modelar el proceso cerebral humano para clasificar imágenes de expresiones faciales por medio de una de las técnicas de *Deep Learning*, logrando así que una máquina sea capaz de aprender de imágenes de expresiones faciales suministradas de ejemplo (datos de entrenamiento) con el objetivo de poder clasificar ejemplos futuros sin ningún tipo de intervención humana en el proceso. En la actualidad, gracias a las Redes Neuronales Convolucionales, se están logrando buenos resultados en la clasificación de imágenes, detección de objetos, comprensión de escena, en comparación con técnicas convencionales, por lo cual en este proyecto se usó la arquitectura de una Red Neuronal Convolutional para clasificar las expresiones faciales, clasificándolas en 6 categorías: enojo, miedo, alegría, tristeza, sorpresa y neutro. Este trabajo aporta a una mejor comprensión en las redes neuronales Convolucionales aplicada al reconocimiento de expresiones faciales e imágenes en general, también ayudara en el desarrollo de futuros proyectos que necesiten del reconocimiento de expresiones faciales, como: estudio de marketing, interacción hombre-máquina, psicología, análisis educativo y otros.

---

**Palabras Claves:** Expresión Facial, *Deep Learning*, Convolutional Neural Networks, Visión por Computador, Reconocimiento de Patrones, Detección de Objetos.

# Abstract

Facial expressions are a means of nonverbal communication that show emotions of people, these expressions help interpersonal transmit information and facilitate the understanding of the meaning of spoken language. So that we believe that to determine the facial expression would be a rich source of information for later use. The objective of this project is to simulate the behavior our brains to recognize images of facial expressions using one of the techniques of Deep Learning, achieving a machine can learn from images of facial expressions supplied sample (training data) in order to classify future examples, without any human intervention in the process. Nowadays thanks to the technique used in this project (convolutional neural network) researchers are achieving good results in image recognition, object detection, understanding scene, compared with other conventional techniques, so in this project use the basic architecture of a convolutional neural network to recognize facial expressions, and classified into 6 categories: happiness, sadness, joy, fear, anger and surprise. This paper give us more understanding on convolutional neural network applied to the recognition of facial expressions and images in general also help in the development of future project requiring the recognition of facial expressions like systems man-machine interaction, marketing analysis based on the facial expressions of people, behavioral studies, mental health and cognitive processes.

---

**Key Words:** Facial Expression Recognition, Understanding Scene, Object Detection, Convolutional Neural Network, Deep Learning.

# Índice General

Indice de Figuras	VIII
Indice de Tablas	X
<b>I Aspectos Generales</b>	<b>3</b>
<b>1. Aspectos Generales</b>	<b>4</b>
1.1. Aspectos Generales . . . . .	4
1.1.1. Descripción del Problema . . . . .	4
1.1.2. Identificación del Problema . . . . .	5
1.2. Antecedentes . . . . .	5
1.2.1. Técnicas Tradicionales para el reconocimiento de expresiones faciales . . . . .	5
1.2.2. Técnicas de <i>Deep Learning</i> . . . . .	6
1.3. Objetivos . . . . .	7
1.3.1. Objetivo General . . . . .	7
1.3.2. Objetivos Específicos . . . . .	7
1.4. Alcances . . . . .	8
1.5. Justificación . . . . .	8
1.6. Metodología . . . . .	9
1.7. Limitaciones . . . . .	9
1.8. Cronograma de Actividades . . . . .	10
<b>II Marco Teórico</b>	<b>11</b>
<b>2. Marco Teórico</b>	<b>12</b>
2.1. Conceptos de Visión . . . . .	12
2.1.1. Visión Humana . . . . .	12
2.1.2. Visión por Computador . . . . .	13
2.2. Detección de Rostros . . . . .	14
2.2.1. Haar Cascade . . . . .	14
2.3. Redes Neuronales . . . . .	15
2.3.1. Biológicas . . . . .	15
2.3.2. Artificiales . . . . .	16

2.4.	Arquitectura de una Red Neuronal Artificial . . . . .	18
2.4.1.	Capas . . . . .	18
2.4.2.	Funciones de Activación . . . . .	19
2.4.3.	Bias o Sesgo . . . . .	21
2.5.	Implementación de una Red Neuronal Artificial . . . . .	22
2.6.	Backpropagation . . . . .	22
2.7.	Deep Learning . . . . .	23
2.8.	Redes mas Comunes Consideradas dentro del Deep Learning . . . . .	25
2.8.1.	Autoencoder . . . . .	25
2.8.2.	Redes Neuronales Recurrentes . . . . .	26
2.8.3.	Redes Neuronales Convolucionales . . . . .	27
2.9.	Arquitectura de una Red Neuronal Convolutacional . . . . .	27
2.9.1.	Capa de Convolución . . . . .	28
2.9.2.	Submuestreo . . . . .	30
2.9.3.	Capa de normalización . . . . .	32
2.9.4.	Capa totalmente conectada . . . . .	32
2.9.5.	Función de normalización(Softmax) . . . . .	33
2.10.	Entrenamiento de una Red Convolutacional . . . . .	33
2.11.	Sobre las Expresiones Faciales . . . . .	34
2.11.1.	Paul Ekman . . . . .	34
2.11.2.	Las seis emociones básicas . . . . .	35
2.11.3.	Otras expresiones faciales . . . . .	37
<b>III</b>	<b>Desarrollo del Proyecto</b>	<b>39</b>
<b>3.</b>	<b>Desarrollo del Detector de Rostros y la Arquitectura de Red Neuronal Convolutacional</b>	<b>40</b>
3.1.	Detección de Rostros . . . . .	40
3.2.	Experimentación en la Elección de Parámetros y Capas en la Construcción de la Arquitectura CNN . . . . .	42
3.3.	Arquitectura Propuesta . . . . .	44
3.4.	Descripción de la Capas de la Arquitectura . . . . .	45
3.5.	Parámetros de la Arquitectura . . . . .	48
3.6.	Entrenamiento de la Red . . . . .	49
3.7.	Pruebas al Modelo Creado . . . . .	50
3.8.	Recopilación de Imágenes de Expresiones Faciales. . . . .	50
3.9.	Base de Datos . . . . .	51
3.9.1.	FER2013 . . . . .	51
3.9.2.	CK+ . . . . .	52
3.9.3.	FER2013 - CK+ . . . . .	52
3.10.	Resultados Experimentales . . . . .	53
3.10.1.	FER2013 . . . . .	54
3.10.2.	CK+ . . . . .	55

3.10.3. FER2013 - CK+ . . . . .	57
<b>Resultados</b>	<b>60</b>
<b>Conclusiones</b>	<b>62</b>
<b>Recomendaciones</b>	<b>63</b>
<b>Trabajos Futuros</b>	<b>64</b>
<b>Bibliografía</b>	<b>64</b>
<b>ANEXOS</b>	<b>68</b>
<b>A. Otros conceptos</b>	<b>68</b>
A.1. Matriz de Confusión . . . . .	68
A.2. Machine Learning . . . . .	69
A.3. Aplicaciones de Machine Learning . . . . .	69
A.4. Early Stopping . . . . .	70
<b>B. Testing</b>	<b>71</b>
B.1. Pruebas satisfactorias . . . . .	71
B.2. Pruebas Fallidas . . . . .	74
<b>C. Herramientas</b>	<b>75</b>
<b>D. Glosario</b>	<b>76</b>
<b>E. Acrónimos</b>	<b>77</b>

# Índice de Figuras

2.1.	Estructura de la percepción visual humana . . . . .	13
2.2.	Esquema de las relaciones entre la visión por computadora y otras áreas afines. . . . .	14
2.3.	Detección Cascade . . . . .	15
2.4.	Neurona biológica . . . . .	16
2.5.	Modelo matemático de una red neuronal . . . . .	17
2.6.	Capas de una red neuronal artificial. . . . .	18
2.7.	Gráfica de la función Sigmoide . . . . .	19
2.8.	Gráfica de la función Tangencial . . . . .	20
2.9.	Gráfica de la función RELU . . . . .	21
2.10.	Arquitectura de un RNA incluida el sesgo . . . . .	21
2.11.	Descenso de gradiente. . . . .	23
2.12.	Tipos de aprendizaje. . . . .	24
2.13.	Aprendizaje supervisado. . . . .	24
2.14.	Aprendizaje no supervisado. . . . .	25
2.15.	Arquitectura de una red neuronal Auto-encoder. . . . .	26
2.16.	Arquitectura de una red neuronal Recurrente. . . . .	26
2.17.	Red neuronal Recurrente. . . . .	27
2.18.	Arquitectura de una red neuronal Convolucional. . . . .	28
2.19.	Ejemplo de convolución con un filtro de dimensiones de $2 \times 2$ . . . . .	29
2.20.	Ejemplo de Submuestreo con una ventana de $2 \times 2$ y calculando el promedio .	31
2.21.	Capa totalmente conectada . . . . .	32
2.22.	Arquitectura de una CNN con Softmax . . . . .	33
2.23.	Expresión Facial de Córnea . . . . .	35
2.24.	Expresión Facial de Felicidad . . . . .	36
2.25.	Expresión Facial de Sorpresa . . . . .	36
2.26.	Expresión Facial de Asco . . . . .	36
2.27.	Expresión Facial de Tristeza . . . . .	37
2.28.	Expresión Facial de Miedo . . . . .	37
3.1.	Ejemplo de una imagen de entrada. . . . .	41
3.2.	Proceso de detección de rostro. . . . .	41
3.3.	Ejemplo de rostros con oclusión. . . . .	41
3.4.	Arquitectura gráfica del modelo propuesto. . . . .	45
3.5.	Imágenes de salida después de la primera convolución. . . . .	46
3.6.	Imágenes de salida después del primer sub-muestreo. . . . .	46

3.7. Imágenes de salida después de la segunda convolución. . . . .	47
3.8. Imágenes de salida después del segundo sub-muestreo. . . . .	47
3.9. Secuencia de pasos en la fase de pruebas. . . . .	50
3.10. Ejemplo de un rostro real y caricaturizado. . . . .	51
3.11. Imágenes de la base de datos FER2013 . . . . .	52
3.12. Imágenes de la base de datos CK+ . . . . .	52
3.13. Precision y recall. . . . .	53
3.14. Matriz de confusión, precisión del Test - FER2013 . . . . .	54
3.15. Precisión durante el proceso de entrenamiento y prueba (%) – FER2013 . . . . .	55
3.16. Matriz de confusión, precisión del Test - CK+ . . . . .	56
3.17. Precisión durante el proceso de entrenamiento y prueba (%) - CK+ . . . . .	56
3.18. Matriz de confusión, precisión del Test FER2013 - CK+ . . . . .	57
3.19. Precisión durante el proceso de entrenamiento y prueba (%) FER2013 - CK+ . . . . .	58
 A.1. Representación gráfica de Early Stopping <i>Fuente: deeplearning4j [7]</i> . . . . .	70
 B.1. Test 1 . . . . .	71
B.2. Test 2 . . . . .	71
B.3. Test 3 . . . . .	72
B.4. Test 4 . . . . .	72
B.5. Test 5 . . . . .	72
B.6. Test 6 . . . . .	72
B.7. Test 7 . . . . .	73
B.8. Test 8 . . . . .	73
B.9. Test 9 . . . . .	74
B.10. Test 10 . . . . .	74
B.11. Test 11 . . . . .	74
B.12. Test 12 . . . . .	74

# Índice de Tablas

1.1. Cronograma de actividades . . . . .	10
3.1. Evaluación de la arquitectura 1 y sus parámetros, FER2013 . . . . .	42
3.2. Evaluación de la arquitectura 1 y sus parámetros, CK+ . . . . .	42
3.3. Evaluación de la arquitectura 2 y sus parámetros, FER2013 . . . . .	43
3.4. Evaluación de la arquitectura 2 y sus parámetros, CK+ . . . . .	43
3.5. Evaluación de la arquitectura 3 y sus parámetros, FER2013 . . . . .	43
3.6. Evaluación de la arquitectura 3 y sus parámetros, CK+ . . . . .	43
3.7. Arquitectura del modelo propuesto . . . . .	44
3.8. Número de parámetros de la arquitectura propuesta. . . . .	49
3.9. Resultados obtenidos - FER2013 . . . . .	54
3.10. Resultados obtenidos - CK+ . . . . .	55
3.11. Resultados obtenidos - FER2013 - CK+ . . . . .	57
A.1. Estructura general de la matriz de confusión . . . . .	68

# Introducción

Uno de los rasgos psicológicos naturales de los seres humanos es la necesidad de interacción entre ellos. La interacción puede darse a través de la comunicación. Propio de éste, es generada y obtenida información que ellos usan para tomar decisiones durante el proceso de comunicación. Actualmente, gracias al gran avance tecnológico vivimos en una era digital que, posibilitó la trascendencia de la comunicación directa a la comunicación indirecta masiva a través de las redes sociales, videoconferencias, etc. Producto de ello, la cantidad de datos (imágenes, videos, textos, etc) generada por minuto en internet es exorbitante. Surgiendo muchas potenciales aplicaciones (reconocimiento de expresiones faciales, reconocimiento de objetos, detección de objetos, etc) que implican el análisis de imágenes y/o videos por computador.

Las expresiones faciales juegan un rol importante en la comunicación no verbal entre los seres humanos, además de ser el medio por el que se transmite más del 55 % de información en el proceso de comunicación. Para el reconocimiento de las expresión facial por medio del computador son diseñados modelos basados en la apariencia y modelos geométricos del rostro humano. Así, en este trabajo abordamos el reconocimiento de expresiones faciales como un problema de clasificación de imágenes de rostros humanos clasificados en 6 categorías (alegre, neutro, feliz, triste, enojado, sorpresa) basados en la apariencia del rostro.

La metodología propuesta para abordar el problema de reconocimiento de expresiones faciales en imágenes es compuesta por 3 partes. La primera parte, comprende la estandarización y la normalización de las imágenes. En la estandarización, las imágenes son redimensionadas a un tamaño estandar y convertidas a tonos de cinza. La normalización es aplicada para reducir la varianza en las imágenes. Para ambos casos se usa métodos y técnicas de procesamiento de imágenes. En la segunda parte, se extrae las características del rostro y se clasifica en las diferentes clases de expresiones faciales (alegre, neutro, feliz, triste, enojado, sorpresa). Para ello, se diseña un conjunto de arquitecturas de redes neuronales convolucionales (CNNs) con diferentes parámetros. Finalmente, se evalua el modelo con mayor rendimiento en 3 bases de datos públicos de expresiones faciales (FER2013, CK+) y un adicional creado a partir de la unión de los dos antes mencionados (Fer2013-CK+).

Adicionalmente, para comprobar el funcionamiento del modelo en imágenes del mundo real, un pequeño conjunto de imágenes de internet es seleccionado por nosotros. Estas imágenes poseen contenidos variados desde imágenes con ruido, rostros con oclusión, fondo complejo hasta imágenes de animes. Primero, es extraido el rostro humano via el detector de rostros *haar cascade* y posteriormente es reconocido su correspondiente expresión facial usando el modelo creado anteriormente.

El desarrollo del presente trabajo puede resumirse en 3 partes.

- **Parte I:** Cubre los aspectos generales del problema, describiendo de una manera detallada el problema al cual se quiere dar solución, los trabajos relacionados, los objetivos a alcanzar, la metodología y las limitaciones encontradas en el desarrollo de la investigación.
- **Parte II:** Proporciona los fundamentos teóricos necesarios que son vitales para el desarrollo y entendimiento del proyecto.
- **Parte III:** Desarrolla y muestra los experimentos realizados con diferentes configuraciones sobre una arquitectura de red neuronal convolucional(CNN). Se describe a detalles el funcionamiento de los métodos elegidos para la detección y el reconocimiento de expresiones faciales. Los resultados obtenidos son interpretados en términos de una métrica de error y precisión, y se proponen trabajos futuros.

# **Parte I**

## **Aspectos Generales**

# Capítulo 1

## Aspectos Generales

### 1.1. Aspectos Generales

#### 1.1.1. Descripción del Problema

Uno de los sentidos más importantes de los seres humanos es la visión. Ésta es empleada para obtener la información visual del entorno físico. Según Aristóteles, “Visión es saber que hay y donde mediante la vista”. De hecho, se calcula que más del 75 % de las tareas del cerebro son empleadas en el análisis de la información visual. El refrán popular de “Una imagen vale más que mil palabras” tiene mucho que ver con los aspectos cognitivos de la especie humana. Casi todas las disciplinas científicas emplean utilajes gráficos para transmitir conocimiento<sup>1</sup>.

Uno de los más grandes concursos a nivel mundial en clasificación de imágenes reporta que las técnicas tradicionales (como las técnicas de extracción de características en imágenes estáticas: *Principal component analysis* (PCA), *Edges detector*, *Gabor wavelet*; Video: PCA, *Discrete cosine transform* (DCT), *optical flow* e *image difference*) están siendo superadas por técnicas de *Deep Learning* que son basadas en el proceso cerebral humano. Dicho éxito se debe a que las técnicas tradicionales requieren de un ambiente controlado y no son tolerables a cambios como: traslación, rotación y escalado. Por otro lado las técnicas de *Deep Learning* demuestran ser mas robustas y efectivas frente a estos tipos de cambios<sup>2</sup>.

Diversas actividades cotidianas necesitan del reconocimiento de imágenes, tal es el caso del reconocimiento de expresiones faciales, que en los últimos años se ha convertido en una de las tareas más estudiadas por investigadores en todo el mundo, con el fin de alcanzar un margen de error mínimo para posteriormente centrarse en el desarrollo de aplicaciones en distintos campos como: estudio de marketing, interacción hombre-computador, psicología y análisis educativo <sup>3</sup>, las cuales han sido abordada por diferentes técnicas tradicionales no obteniendo resultados prometedores en imágenes reales que contienen distintos tipos de variaciones (mencionados en el párrafo anterior). Limitando así, la implementación y desarrollo de aplicaciones útiles para el bien común (aplicaciones antes mencionadas).

---

<sup>1</sup>Visión Humana, fuente: Sistemas Adaptativos y Bioinspirados en Inteligencia Artificial(S.A.B.I.A.)

<sup>2</sup>*Deep Learning vs. Machine Learning* fuente: Analytics Vidhya

<sup>3</sup>Las expresiones faciales de las emociones, historia y aplicaciones, fuente: Ciencia Cognitiva

### **1.1.2. Identificación del Problema**

Las técnicas tradicionales para el reconocimiento de expresiones faciales usadas en la actualidad necesitan de un ambiente controlado (iluminación constante, alta calidad de imagen, poco ruido, imagen sin oclusión), y no son tolerables a cambios como rotación, traslación, escalado. Limitando así, la creación de aplicaciones con imágenes del mundo real (imágenes obtenidas a partir de cámaras de seguridad). Por lo que hay la necesidad de usar nuevas técnicas del estado del arte que nos permitan obtener mejores resultados superando así la limitación antes mencionada.

## **1.2. Antecedentes**

Se muestra una lista de trabajos resaltantes que hacen uso de técnicas de *deep learning*, los cuales sirvieron de inspiración y fuente de información valiosa en el desarrollo de este trabajo. También se presentan trabajos dedicados al reconocimiento de expresiones faciales utilizando técnicas tradicionales de visión por computador y *machine learning*.

### **1.2.1. Técnicas Tradicionales para el reconocimiento de expresiones faciales**

Muchos abordajes fueron propuestos para la tarea de reconocimiento de expresiones faciales basados técnicas tradicionales y *machine learning*.

#### **1. “Learning active facial patches for expression analysis” [41]**

Zhong et al. propone un método para el reconocimiento de expresiones faciales basado en *patches* informativos de expresiones faciales (e.g. ojos, mejillas, boca, etc) de los rostros humanos contenidos en las imágenes. Así, introduce un marco de aprendizaje por tareas multiples (MTSL) de dos etapas para ubicar de manera eficiente los *patches* discriminativos en las imágenes con contenido facial. La primera etapa consiste en extraer un conjunto de *patches* por cada expresión facial dentro de un conjunto de datos de entrenamiento. En la segunda parte del MTS defense, extraído el conjunto de *patches* por cada expresión facial en el conjunto de datos de entrenamiento dos tareas son desarrolladas, el reconocimiento de expresión facial y verificación facial. Estas dos tareas son integrados para aprender de los *patches* obtenidos con anterioridad. El aprendizaje tiene como objetivo determinar cuáles son los específicos *patches* faciales que corresponden a cada expresión facial. Finalmente, un clasificador SVM es entrenado para reconocer las características contenidas en los *patches* para una de las seis expresiones faciales consideradas en este trabajo.

#### **2. “Local gabor binary patterns from three orthogonal planes for automatic facial expression recognition” [1]**

Almaev et al. propone un descriptor dinámico de apariencias *Local Gabor Binary Patterns de Three Orthogonal Planes*(LGBP-TOP) donde, analizando la textura espacial

y dinámica, y combinado con filtros de gabor logra un alto nivel de precisión en el reconocimiento de expresiones faciales en tiempo real. Además, el descriptor propuesto es robusto a errores en la alineación rotacional propios de la captura de registros.

### 3. “An integrated approach for efficient analysis of facial expressions” [13]

Ghayoumi et al. propone la integración de tres métodos para el reconocimiento de expresiones faciales. *Locality Sensitive Hashing* (LSH), *Principal Component Analysis* (PCA) y *Linear Discriminant Analysis* (LDA) son los tres métodos utilizados para el reconocimiento de expresiones faciales en este trabajo. Para reconocer una expresión facial son extraídos los vectores de características de dos regiones más informativas del rostro, la región de los ojos y la región de la boca. El LSH esta compuesto por funciones hash los cuales son usados para mapear los vectores de características extraídos de las imágenes en bloques de colisiones, donde cada bloque esta registrado como la ubicación de vectores de características pertenecientes a una específica expresión facial a priori. Así, se logra reducir redundancia en el espacio de representación de las imágenes respecto a las imágenes que pertenecen al mismo tipo de expresión facial. Luego de este paso son aplicados los métodos de reducción de dimensionalidad PCA y LDA para aliviar la complejidad computacional y reducir redundancia en los datos. Por último, es entrenado un clasificador SVM con los datos obtenidos después de aplicado los métodos de reducción de dimensionalidad.

#### 1.2.2. Técnicas de *Deep Learning*

Muchos de los avances en visión por computador, procesamiento del lenguaje natural, bioinformática y otros campos de investigación se debe a la utilización de métodos y técnicas de inteligencia artificial, específicamente técnicas de *machine learning* y *deep learning*. Este último, es el causante de una gran impacto en el avance tecnológico con relacionamiento directo en el mundo comercial (p.e. carros autónomos, chatbots, creación de nuevos fármacos por computador, etc).

### 1. “Gradient-based learning applied to document recognition” [20]

Yan Lecun et al. introduce uno de los primeros trabajos con redes neuronales convolucionales, un tipo especial de *deep learning*. En este trabajo, muestra la potencialidad de la técnica de aprendizaje basado en gradiente. Esta técnica es utilizado en el entrenamiento de una *multilayer perceptron* via el algoritmo de *backpropagation*. Es diseñado una red neuronal convolucional para tratar con la variabilidad de forma 2D en imágenes para el reconocimiento de dígitos escritos a mano. También es introducido dos sistemas on-line para el reconocimiento de documentos escritos a mano. Para controlar los componentes del sistema tales como la extracción, segmentación, reconocimiento y modelado del lenguaje; es introducido un nuevo paradigma de aprendizaje llamado *Graph Transformer networks*. Además, es descrito la potencialidad del *Graph transformer networks* aplicado a la lectura de cheques de banco. Donde, las redes neuronales convolucionales para el reconocimiento de caracteres combinado con técnicas de entrenamiento global proporcionaron un alto rendimiento. Esto fue demostrado en el área

comercial al ser capaz el modelo de leer cheques personales de forma automatizada, logrando leer millones de cheques por día.

## 2. “Imagenet classification with deep convolutional neural networks” [19]

Krizhevsky et al. propone una arquitectura de red neuronal convolucional profunda que es entrenado para clasificar 1.2 millones de imágenes en alta resolución en 1000 categorías. La red propuesta en este trabajo llega a estar compuesto por 60 millones de parámetros y 650000 neuronas. Para el entrenamiento fueron usados *GPUs* para acelerar el computo de multiplicaciones matriciales requeridas en las operaciones de convolución. Este trabajo logra el estado del arte en clasificación de imágenes a gran escala en la competición ILSVRC-2012. Esto es demostrado al lograr reducir el error de clasificación de imágenes en la base de datos ImageNet de 26.2 % a 15.3 %. Siendo así, uno de los trabajos con más impacto en visión por computador.

Así, es demostrado que una de las principales desventajas de los métodos basados en extracción de características diseñadas a mano es el tiempo de computo lo que les dificulta ser escalables, requisito principal para aplicaciones en el mundo real. además, estos métodos solo cubren una parte específica de casos, esto debido a las suposiciones que se toman para casos específicos que no logran cubrir todo el espectro de posibilidades para poder crear modelos generalizables. Por otro lado, los métodos de *deep learning* demostraron ser capaces de crear modelos generalizables y escalables. Hecho este análisis, fuimos motivados a desarrollar una arquitectura de red neuronal convolucional para el reconocimiento de expresiones faciales.

## 1.3. Objetivos

### 1.3.1. Objetivo General

Desarrollar una arquitectura de red neuronal convolucional que sea capaz de obtener niveles de precisión confiables (mínimo margen de error) en el reconocimiento de expresiones faciales, permitiendo así contribuir en el desarrollo de futuras aplicaciones del mundo real que sirvan para el beneficio de la sociedad.

### 1.3.2. Objetivos Específicos

- Selección de las bases de datos de expresiones faciales y el pre-procesamiento de estos datos (estandarización y normalización).
- Investigar los filtros de convolución para la correcta selección de los parámetros.
- Investigar la función de submuestreo para la correcta selección de los parámetros.
- Investigar las funciones de activación y funciones de normalización para la correcta selección de los parámetros.

- Diseñar la arquitectura propuesta (configuración de parámetros, número de capas y funciones de activación y normalización), basandonos en los objetivos previos.
- Entrenar la arquitectura propuesta.
- Evaluar el modelo creado a partir de la arquitectura propuesta.
- Analizar e interpretar los resultados

## 1.4. Alcances

En este trabajo de investigación se lograron los siguientes alcances.

- Se propuso una nueva arquitectura para el reconocimiento de expresiones faciales, basada en las redes neuronales convolucionales. El modelo creado fue capaz de obtener altos niveles de precisión que serán de utilidad para el desarrollo de futuras aplicaciones en el mundo real.
- Se creó una nueva base de datos, la cual fue resultado de la unión de las dos bases de datos antes mencionadas(FER2013 y CK+).
- Contribuimos con la comunidad académica del país y la región brindandoles información de un tema de investigación actual que servirá como base para el desarrollo de futuras aplicaciones y trabajos relacionados.

## 1.5. Justificación

En la actualidad se ha dado más realce a algunas disciplinas de la inteligencia artificial como: *machine learning* y *deep learning*, disciplinas que brindan distintas técnicas que están dando solución a problemas de clasificación de imágenes, comprensión de escena, análisis de sentimientos y otros. Así, es el caso de la visión artificial donde las redes neuronales convolucionales están proporcionando mejores resultados en comparación con algoritmos y técnicas tradicionales.

En este trabajo, presentamos un estudio resumido de la investigación hecha en *deep learning* con aplicación en el reconocimiento de expresiones faciales que servirá tanto para los investigadores como para los lectores. También este trabajo ayudara para el desarrollo de futuros proyectos de clasificación de imágenes en distintos campos (seguridad, medicina y biología, internet y la nube, entretenimiento, máquinas autónomas y otros)<sup>4</sup>.

---

<sup>4</sup>Aplicaciones de Deep Learning, fuente: NVIDIA GPUs - el motor del aprendizaje profundo(deep learning)

## 1.6. Metodología

Dada la naturaleza del trabajo de investigación, se utilizó los métodos de investigación bibliográfica, explorativa y aplicativa. Bibliográfica ya que se recogió y analizó información para obtener conocimientos previos sobre *deep learning* y el detector *haar cascade*. Explorativa porque se seleccionó información relevante procedente de la etapa de investigación bibliográfica, que sirvió para la construcción de la arquitectura de una *red neuronal convolucional* basándonos en trabajos previos relacionados con la línea de investigación. Aplicativa por que se utilizaron los conocimientos adquiridos [28] [30].

## 1.7. Limitaciones

- Difícil acceso a herramientas tecnológicas de hardware, principalmente GPU's de alta capacidad, necesarias para la fase de entrenamiento de la red neuronal convolucional.
- Difícil acceso a información científica de fuentes confiables (IEEE, ACM, Springer, etc). Esto debido al elevado costo para acceder a ellos.
- Carencia de organizaciones peruanas que brinden grandes bases de datos de expresiones faciales para poder utilizarlos en la fase de entrenamiento y crear un modelo para el reconocimiento de expresiones faciales de la región o del país. Así, fuimos llevados a utilizar bases de datos de organizaciones extranjeras que fomentan la investigación en esta área.

## 1.8. Cronograma de Actividades



Tabla 1.1: Cronograma de actividades

# **Parte II**

## **Marco Teórico**

# Capítulo 2

## Marco Teórico

### 2.1. Conceptos de Visión

#### 2.1.1. Visión Humana

De una manera muy general, visión se entiende como toda acción de ver, sin embargo, desde un punto de vista mas técnico, visión es la capacidad de interpretar nuestro entorno gracias a los rayos de luz que alcanzan el ojo. Otros autores definen visión como una capacidad necesaria mas no imprescindible para realizar las actividades cotidianas.

Desde el punto de la vista de la medicina, la visión humana o sentido de la vista se reduce a un organo receptor conocido como el *ojo*, la membrana y retina son los encargados de recibir las impresiones luminosas para luego transmitirlas al cerebro por medio del nervio óptico (ver figura 2.1). En adición, el ojo es un órgano situado en la cavidad orbitaria, esta protegida por los párpados y por la secreción de las glándulas lagrimales. Los ojos son sensibles a ondas de radiación electromagnética de longitudes específicas. Estas ondas se registran como la sensación de la luz. Cuando la luz penetra en el ojo, pasa a través de la córnea, la pupila y el cristalino, y llega por último a la retina, donde la energía electromagnética de la luz se convierte en impulsos nerviosos que pueden ser utilizados por el cerebro. Los impulsos abandonan el ojo a través del nervio óptico. La región más sensible del ojo en la visión normal diurna es una pequeña depresión de la retina llamada fóvea en el cual se enfoca la luz que viene del centro del campo visual (por campo visual entendemos aquello a lo que mira el sujeto). Puesto que la lente simple convexa invierte la imagen, el campo visual derecho es representado a la izquierda de la retina y el campo inferior representado en lo alto de la retina. El ojo es un sistema óptico muy imperfecto. Las ondas de luz no solo tienen que pasar a través de los humores y el cristalino, después penetrar la red de los vasos sanguíneos y fibras nerviosas antes de que lleguen las células sensibles los bastones y los conos de la retina donde la luz se convierte en impulsos nerviosos. A pesar de estas imperfecciones el ojo funciona muy bien. La fóvea es capaz de percibir un cable telefónico a 400 m de distancia. En buenas condiciones el ojo puede percibir un alambre cuyo grosor no cubre más de 0,5 mm.

Tambien existen otras definiciones que indican que, el ojo es la puerta de entrada por la que ingresan los estímulos luminosos que se transforman en impulsos eléctricos gracias a unas células especializadas de la retina que son los conos y los bastones. Entonces, el nervio óptico

transmite los impulsos eléctricos generados en la retina al cerebro, donde son procesados en la corteza visual. Finalmente, en el cerebro tiene lugar el complicado proceso de la percepción visual gracias al cual somos capaces de percibir la forma de los objetos, identificar distancias, detectar los colores y el movimiento [2].

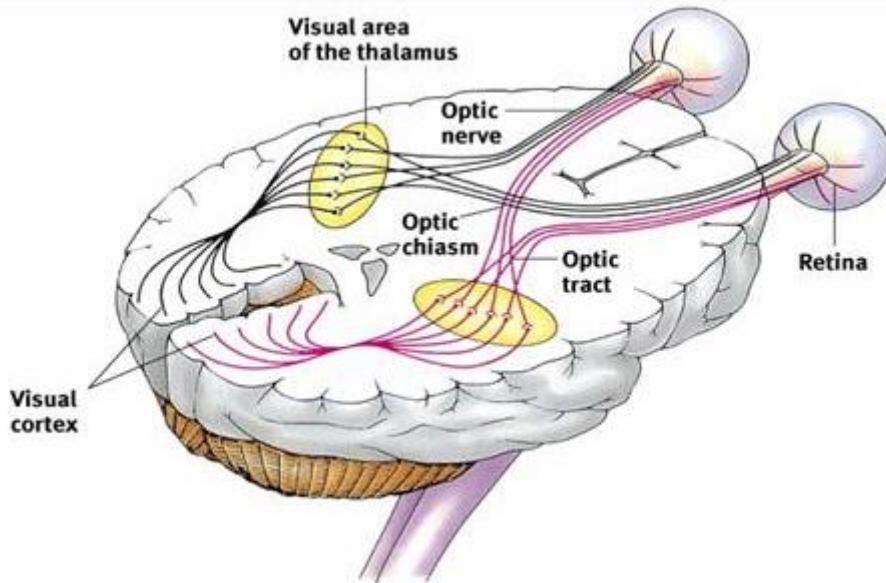


Figura 2.1: Estructura de la percepción visual humana.

Fuente: Fernando Vila Arroyo, “El Libro Blanco de la Iluminación”. España 2013.

### 2.1.2. Visión por Computador

La visión artificial o también conocida como visión por computador es una disciplina científica que incluye métodos para adquirir, procesar, analizar y comprender las imágenes del mundo real con el fin de producir información numérica o simbólica para que puedan ser tratados por un computador. Tal y como los humanos usamos nuestros ojos y cerebros para comprender el mundo que nos rodea, la visión por computador trata de producir el mismo efecto para que las computadoras puedan percibir y comprender una imagen o secuencia de imágenes y actuar según convenga en una determinada situación. Esta comprensión se consigue gracias a distintos campos como la geometría, la estadística, la física y otras disciplinas. La adquisición de los datos se consigue por varios medios como secuencias de imágenes, vistas desde varias cámaras de video o datos multidimensionales desde un escáner médico.

Hay muchas tecnologías que utilizan la visión por computador (figura 2.2), entre las cuales tenemos: reconocimiento de objetos, detección de eventos, reconstrucción de una escena (*mapping*) y restauración de imágenes [39].

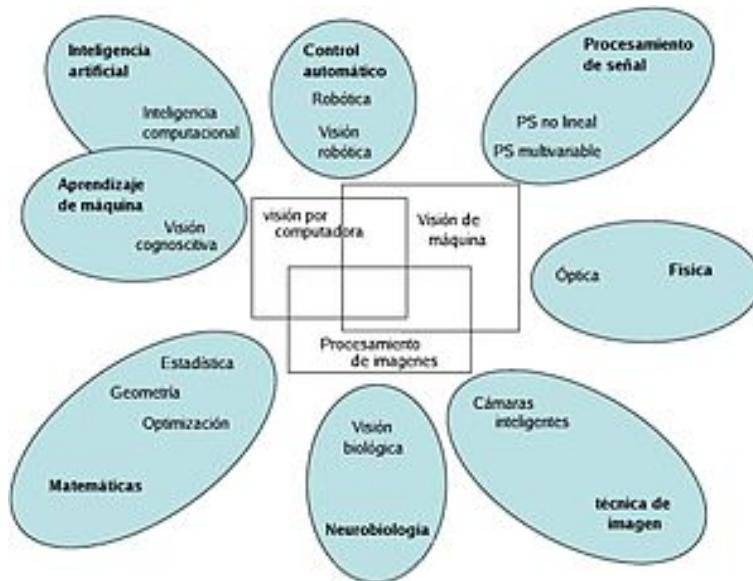


Figura 2.2: Esquema de las relaciones entre la visión por computadora y otras áreas afines.

Fuente: *Visión artificial, Wikipedia*.

## 2.2. Detección de Rostros

En los últimos años investigadores de todo el mundo vienen trabajando en la detección y reconocimiento de rostros debido a la gran cantidad de aplicaciones que este brinda, aplicaciones relacionadas con la seguridad pública, estudios de marketing y psicología de las personas. El rostro es una de las partes del cuerpo que más rasgos representativos muestra en una persona, por lo cual es de suma importancia poder identificar, reconocer y clasificar cada una de estas características, de ahí su gran importancia para el desarrollo de las aplicaciones antes mencionadas. En la localización o detección de rostros, la primera etapa de los sistemas automatizados basados en visión por computador es encontrar el área que envuelve el rostro dentro de la imagen de entrada. La ubicación exacta de la cara es todavía una tarea difícil. Dentro de los muchos trabajos orientados a resolver este problema, Viola-Jones [33] ha sido ampliamente utilizada debido a su robustez para la localización de objetos. La segunda etapa relacionada con el reconocimiento o clasificación de rostros es otra tarea muy estudiada. En general para realizar esta tarea son utilizados algoritmos de clasificación de imágenes que están disponibles en muchas librerías *open source*, tales como OpenCV [23].

### 2.2.1. Haar Cascade

*Haar Cascade* es un efectivo método para la detección de objetos, basado en la utilización de características de tipo *haar* este método resulta muy eficiente y robusto para este tipo de tareas. Fue propuesto por Paul Viola y Michael Jones [33], de ahí el sobrenombre Viola-Jones. Debido a que este método sirve para la detección de objetos, investigadores vieron por conveniente aprovechar sus grandes cualidades para la detección de rostros, alcanzando

así altos niveles de precisión en esta tarea. Existen muchas librerías *open source* que disponibilizan modulos con la implementación de este algoritmo, una de esas tantas es la popular librería *Open Computer Vision Library* más conocida como OpenCV, el *framework* general de detectores de objetos se ha popularizado y ha motivado a la comunidad a generar sus propios clasificadores de objetos. Estos clasificadores usan características parecidas a las del tipo *haar* que se aplican sobre la imagen. [23].

*Haar Cascade* de una manera resumida, es un enfoque basado en *machine learning*. Donde, una función *cascade* es entrenada con muchas imágenes positivas (imágenes de caras) y negativas (imágenes que no sean caras). Entonces, se extraen las características de ellas, entrenando un modelo capaz de reconocer rostros con dichas características.

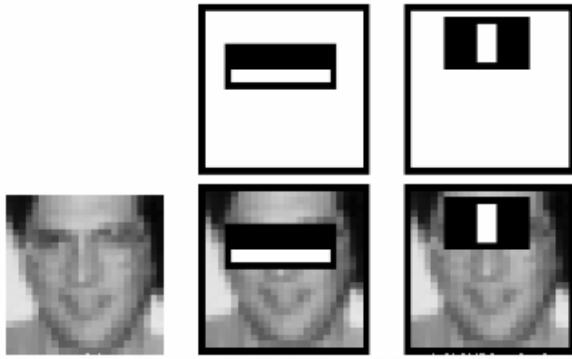


Figura 2.3: Detección Cascade.

Fuente: *Evaluation of Haar Cascade Classifiers for Face Detection, OpenCV*.

## 2.3. Redes Neuronales

### 2.3.1. Biológicas

Son el principal elemento del sistema nervioso. Las redes neuronales biológicas son el resultado de la unión de varias neuronas entrelazadas entre sí. Una neurona es una célula compuesta por tres partes fundamentales: el cuerpo, un número de extensiones llamadas dendritas que sirven de entradas, y una larga extensión llamada axón, la cual se activa como salida. Existe un proceso de comunicación entre neuronas, el cual es conocido como "la sinapsis", este proceso conecta el axón de una neurona a las dendritas de las otras neuronas para comunicarse por medio de impulsos eléctricos. Las neuronas están dispuestas en múltiples capas. Por lo general, las neuronas de una primera capa reciben entradas desde otra capa y envían sus salidas o impulsos nerviosos a las neuronas de una tercera. Existe un proceso de retroalimentación que se origina cuando los impulsos nerviosos de una neuronal son enviadas a ella misma, originando así un ciclo donde la información se mantiene por períodos de tiempo. Similar, puede ocurrir la comunicación entre neuronas de la misma capa.

Las conexiones entre neuronas tienen pesos asociados que representan la influencia de una sobre la otra. Si dos neuronas no están conectadas, el correspondiente peso de enlace es cero. Esencialmente, cada una envía su información de estado multiplicado por el correspondiente

peso a todas las neuronas conectadas con ella. Luego cada una, a su vez, suma los valores recibidos desde sus dendritas para actualizar sus estados respectivos.

Las redes biológicas son entrenadas por medio de la experiencia vivida durante el día a día (imágenes recolectadas por el ojo humano y procesadas por el cerebro), de esta forma, estimulando a cada neurona a aprender características especiales que ayuden a identificar un determinado objeto. Cabe mencionar que la efectividad y precisión con la que se pueda reconocer un objeto, depende mucho de la cantidad de imágenes previas que se haya visto sobre este. Además, se producirán respuestas cuando, en la utilización, se presenten entradas totalmente nuevas para el sistema. De este modo el sistema de red neuronal no reside necesariamente en la elegancia de la solución particular sino en su generalidad de hallar solución a problemas particulares, habiéndose proporcionado ejemplos del comportamiento deseado. Esto permite la evolución de los sistemas autómatas sin una reprogramación explícita [18].

De acuerdo con estudios previos, muchos trabajos orientados al estudio del cerebro humano y la medicina, mencionan que una persona tiene alrededor de  $10^{11}$  neuronas, cada una con alrededor de  $10^4$  salidas. Donde, la estructura de neuronas de la corteza cerebral es modular: si bien todas las partes del cerebro son relativamente similares, diferentes partes hacen diferentes cosas; a partir de una estructura general, según la experiencia se generan nuevas estructuras específicas al problema a resolver [25].

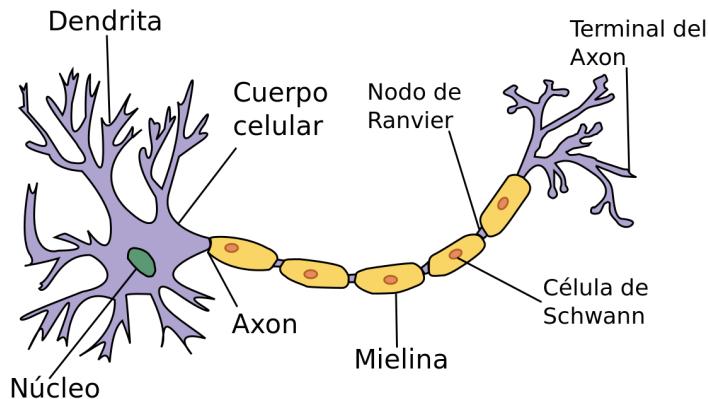


Figura 2.4: Neurona biológica

Fuente: Patri Tezanos, Neurociencia, 2016.

### 2.3.2. Artificiales

Las redes neuronales artificiales (ANN) imitan el funcionamiento del cerebro, básicamente, el tipo de conexiones existentes (ejemplo: las neuronas de una capa previa están conectadas a neuronas de la siguiente capa), estructura (número de capas) y transferencia de información entre neuronas. Este tipo de arquitectura son aptas para resolver problemas que no poseen un algoritmo claramente definido, transformando así una entrada en una salida; aprenden, reconocen y aplican relaciones entre objetos. Para realizar este tipo de procesos o tareas, se emplea normalmente un conjunto de ejemplos representativos para .<sup>e</sup>ntrenar.<sup>e</sup>l sistema o la red neuronal (en nuestro caso, imágenes de expresiones faciales), que, a su vez, se adaptan

ajustando los pesos de cada neurona de tal manera que pueda producir salidas deseadas a cada imagen de entrada.

Además, similar a las redes neuronales biológicas, se producirán respuestas, cuando en la utilización se presenten entradas totalmente desconocidas por el sistema entrenado. De este modo el sistema de red neuronal artificial no se limita a solo reconocer imágenes que hayan sido previamente vistas, sino en su generalidad para hallar similitudes entre la imagen de entrada y las imágenes previas antes de la fase de entrenamiento, generando así un sistema robusto capaz de obtener una semejanza enorme a las redes neuronales biológicas.

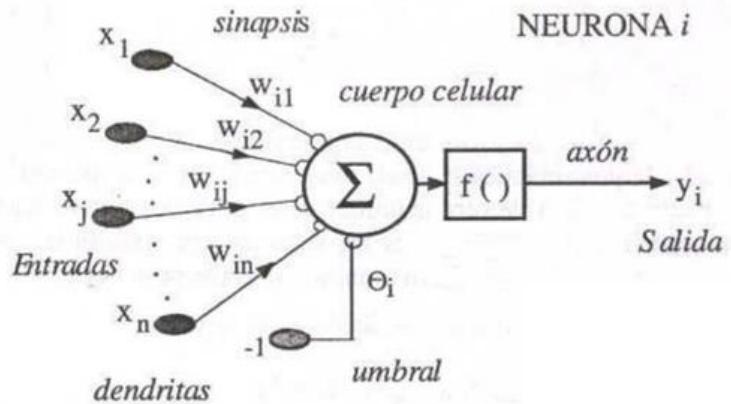


Figura 2.5: Modelo matemático de una red neuronal

Fuente: Yuly Cristina Moreira Monserrate, Inteligencia Artificial, 2015.

Las redes neuronales artificiales se basan en el circuito de procesamiento de entradas en el cual los pesos son sumados. Las funciones de peso son conocidas generalmente como atenuadores. En la implementación, las entradas a una neurona son pesadas multiplicando el valor de la entrada por un factor que es menor o igual a uno. El valor de los factores de peso es determinado por el algoritmo de aprendizaje. Las entradas atenuadas son sumadas usando una función no lineal llamada "Síntesis". Si la salida de la función suma excede el valor de entrada máximo de la neurona, esta responde generando una salida. Cada neurona tiene varias entradas y su salida está conectada a un conjunto de otros procesadores de entradas.

Cuando una red funciona en modo normal, a partir de los datos presentados en la entrada, se genera un patrón específico de salida. La relación entre la entrada y la salida será determinada durante la etapa de entrenamiento, entonces cuando una entrada conocida es presentada se dará una salida efectiva. Durante esta etapa de entrenamiento, el algoritmo de aprendizaje ajusta los pesos de las entradas hasta que se alcanza la salida esperada, esto se logra por medio de la minimización de una función de costo (ver ecuación 2.1), la cual es representada como la diferencia entre la salida esperada y la salida obtenida.

$$Y_i = f(\sum W_{i,j} X_j - \theta_i) \quad (2.1)$$

Debido a la compleja conexión que se realiza entre capas de la red, la complejidad con respecto a términos computacionales y tiempo de ejecución, puede crecer de manera exponencial. Esto se debe a que cada neurona está conectada a cientos de neuronas de la siguiente

capa y cada neurona de esa siguiente capa realiza lo mismo. Por lo tanto, mientras mas sea el numero de capas y neuronas, aparentemente el sistema sera mejor, pero tambien sera mas complejo y dificil de entrenar. [18].

La figura 2.5 muestra las partes de una neurona artificial, la cual es similar a la estructura de una neurona biológica.

## 2.4. Arquitectura de una Red Neuronal Artificial

### 2.4.1. Capas

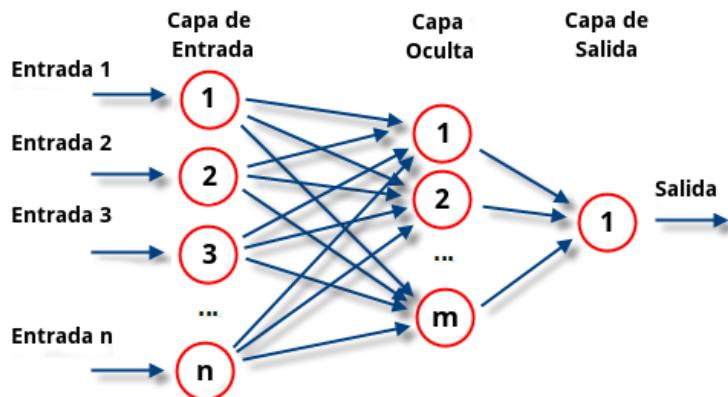


Figura 2.6: Capas de una red neuronal artificial.

Fuente: Perceptrón multicapa, Wikipedia.

La representación matemática de la salida de una neurona, esta dada por la ecuación 2.2. donde  $x$  y  $w$  representan la entrada y pesos de la neurona. La variable  $i$  corresponde al número de pesos y entradas (estos dos siempre tienen que ser iguales en cantidad). Cada peso es multiplicado por su respectiva entrada y el producto de esas multiplicación son alimentadas en una función de activación que es denotada por la letra griega  $\phi$ . La neurona es solo la unidad de una red entera, ya que ésta está compuesta por tres grandes capas:

$$f(x_i, w_i) = \phi\left(\sum_{i=1}^n (w_i * x_i)\right) \quad (2.2)$$

- **Capa de Entrada.-** Es la encargada de recepcionar los datos de entrada que posteriormente serán procesados. La representación de estas estradas puede variar dependiendo del tipo de problema que se quiera resolver. En el caso de imágenes, esta se puede representar como una matriz que contiene un determinado valor representativo para cada pixel.
- **Capa Oculta.-** Es la encargada de realizar el procesamiento pesado de la información, para posteriormente enviar la información ya procesada a la capa de salida.

- **Capa de Salida.-** Contiene los resultados como una lista de números. Donde el número de neuronas de la capa de salida tiene que ser igual al número de posibles salidas que ofrezca el problema a resolver, otorgando un valor a cada neurona y finalmente, por medio de una función de normalización obtener la salida definitiva, que corresponde a una sola neurona.

## 2.4.2. Funciones de Activación

La función de activación es la encargada de mantener los números producidos por cada neurona dentro de una rango razonable (generalmente números reales entre 0 y 1). Esta función recibe como entrada la suma de todos los números que llegan por medio de las conexiones entrantes, transforma el valor mediante una determinada función y produce un nuevo valor que será utilizado para la siguiente iteración. Existen distintas funciones de activación estudiadas y experimentadas en trabajos previos, donde, algunas de ellas muestran mayor desempeño que las otras.

- **Función de activación Sigmoide**

Muchos procesos naturales y curvas de aprendizaje de sistemas complejos muestran una progresión temporal desde unos niveles bajos al inicio, hasta acercarse a un clímax transcurrido un cierto tiempo; la transición se produce en una región caracterizada por una fuerte aceleración intermedia. La función *sigmoide* permite describir esta evolución. Su gráfica tiene una típica forma de "S"(ver figura 2.7), y esta limitada en el rango de 0 a 1 en el eje de las ordenadas. A menudo la función *sigmoide* se refiere al caso particular de la función logística y que viene definida por la ecuación 2.3 [37].

$$f(x) = \frac{1}{1 + \exp^{-x}} \quad (2.3)$$

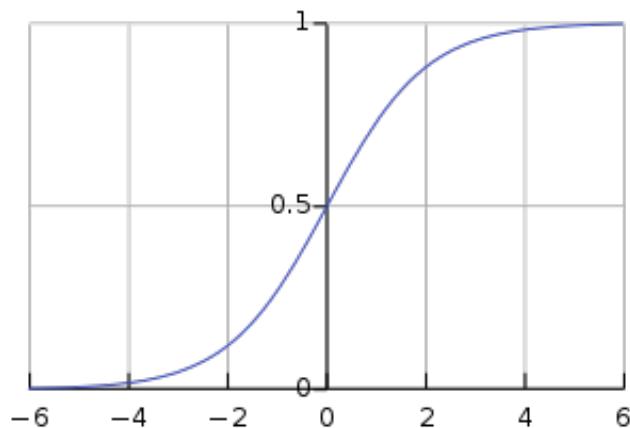


Figura 2.7: Gráfica de la función Sigmoide

Fuente: *Sigmoid Function*, Wikipedia.

### ■ Función de activación Tangencial

Es la versión continua de la función signo y se usa en problemas de aproximación. Es importante por sus propiedades analíticas. Es continua a valores en el intervalo  $[-1,1]$  (ver figura 2.8) e infinitamente diferenciable. Esta función está definida por la ecuación 2.4 [38].

$$\tanh(x) = \frac{\exp^x - \exp^{-x}}{\exp^x + \exp^{-x}} \quad (2.4)$$

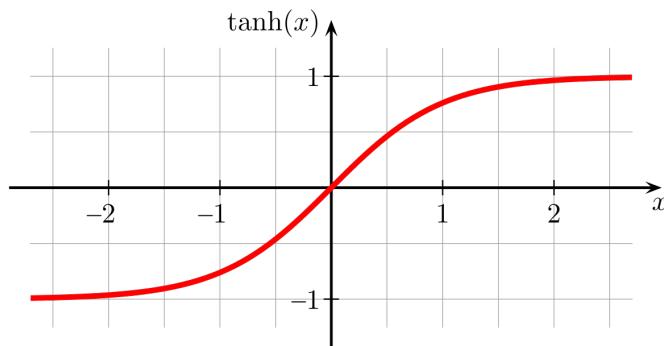


Figura 2.8: Gráfica de la función Tangencial

Fuente: *Tangente hiperbólica*, Wikipedia.

### ■ Función de activación RELU (*Rectified Linear Unit*)

Se conoce como una función de rampa y es análoga a la rectificación de onda media. Esta función de activación fue introducida por primera vez a una red dinámica, en un artículo del año 2000, con fuertes motivaciones biológicas y justificaciones matemáticas. Finalmente en el año 2015, después de ser usada dentro de las redes neuronales convolucionales, alcanzó un gran nivel de eficacia y robustez, superando ampliamente a las funciones de activación *logística sigmoide* (la cual está inspirada en la teoría de probabilidades) y *tangente hiperbólica*. Siendo así la más popular para las redes neuronales profundas. Esta función es representada por medio de la ecuación 2.5, y debido a esa ecuación, la parte negativa en el eje abscisas se mantiene en cero, generando la gráfica como se muestra en la figura 2.9 [36].

$$f(x) = \text{Max}(0, x) \quad (2.5)$$

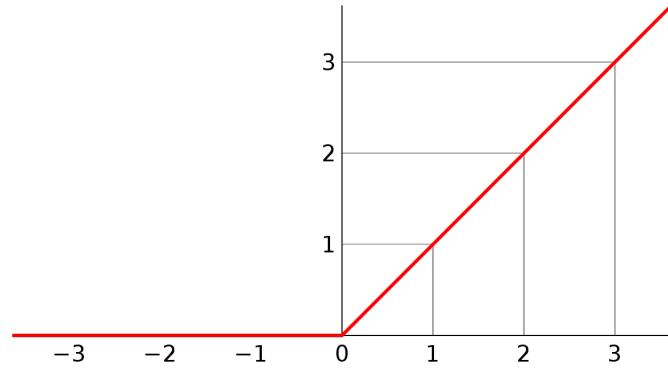


Figura 2.9: Gráfica de la función RELU

Fuente: *Hyperbolic tangent and ReLU neurons*, Int8.

### 2.4.3. Bias o Sesgo

Es una cantidad constante que cada neurona de alguna capa oculta añade justo antes de aplicar la función de activación, este valor puede incrementar o disminuir a la suma de productos (sumatoria de los productos obtenidos después de multiplicar el valor de cada neurona con su respectivo peso que conecta hacia la siguiente neurona). El objetivo de este valor constante es lograr una convergencia más rápida de la red y evitar que el valor final entrante a una neurona sea un valor muy despreciable o muy significativo [14]. La figura 2.10 muestra como este valor es intergado en la arquitectura de una red neuronal.

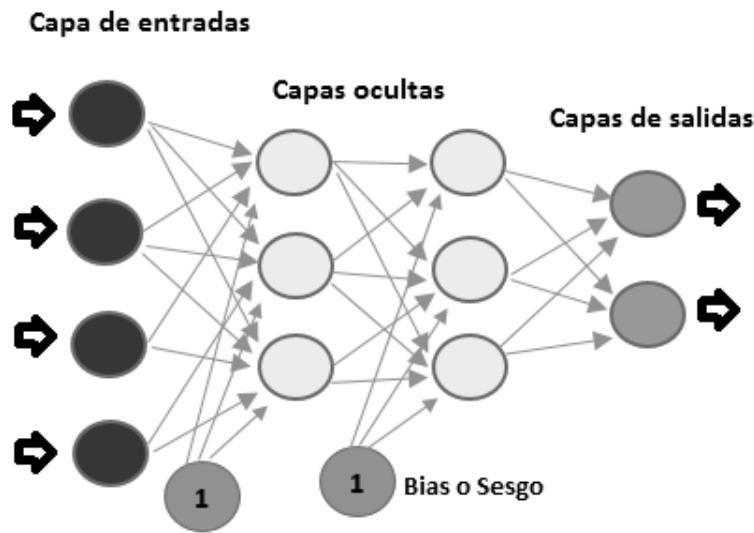


Figura 2.10: Arquitectura de un RNA incluida el sesgo

Fuente: Propio

**Que rol juega los bías?** Las funciones de activación vistas en la sección previa especifican el resultado de una sola neurona. Junto, el peso y el sesgo de una neurona dan forma a la

salida de la activación para producir la salida deseada. La ecuación 2.6 muestra este proceso.

$$f(x, w, b) = \frac{1}{1 + e^{-wx+b}} \quad (2.6)$$

La variable  $x$  representa una entrada simple a la red neuronal. Las variables  $w$  y  $b$  especifican los pesos y bías de la red neuronal. La ecuación de arriba es una combinación de la ecuación 2.2 que especifica una red neuronal y la ecuación 2.3 que representa la función de activación *sigmoide*.

## 2.5. Implementación de una Red Neuronal Artificial

Una forma sencilla de implementar redes neuronales, consiste en almacenar los pesos en matrices. Se considera que la red neuronal es un grafo y simplemente representamos este grafo por medio de su matriz de adyacencia, donde cada posición  $(i, j)$  almacena el peso entre la neurona  $i$  y la neurona  $j$ . Posteriormente guardamos los valores de todas las neuronas de una determinada capa en un vector, el producto del vector y la matriz de pesos de salida nos da los valores de entrada de cada neurona en la siguiente capa. Después se aplica la función de activación a cada elemento de ese segundo vector, y repetimos el proceso [18].

La implementación antes mencionada es una idea general de como se podría implementar una red neuronal (no es el único algoritmo). En la vida real existen distintos tipos de redes neuronales, donde cada una de ellas presenta una implementación diferente, siendo algunas de ellas más eficientes computacionalmente que otras.

## 2.6. Backpropagation

El *BackPropagation* es un algoritmo de aprendizaje supervisado que se usa para entrenar redes neuronales artificial, dicho algoritmo se basa en el descenso de gradiente que es un algoritmo de optimización utilizado para determinar los valores de los parámetros (coeficientes) de una función  $f$  que minimiza una función de costes. El descenso de gradiente se utiliza mejor cuando los parámetros no pueden ser calculados analíticamente (por ejemplo, usando álgebra lineal) y deben ser buscados por un algoritmo de optimización [21].

La figura 2.11 muestra la representación de como el descenso de gradiente trabaja en la búsqueda de los mejores parámetros para una determinada función (ejemplo: sea la función  $f(x) = ax + b$ , donde los parámetros son las constantes  $a$  y  $b$ ). El objetivo es alcanzar el círculo central, el cual representa los parámetros óptimos buscados. Para alcanzar dicho objetivo se realiza una serie de paso (representados por las rectas trazadas entre cada par de puntos rojos, como se muestra en la figura 2.11) que sirven para ajustar los parámetros de tal forma que poco a poco se acerquen a la función objetivo.

El algoritmo de *Backpropagation* es muy intuitivo. Dada una entrada a la red, esta genera una salida, la cual posteriormente es comparada con la salida deseada, consiguiendo así el primer error de salida (función de costo o diferencia entre la salida deseada y la salida obtenida por la red). Seguido, este error es propagado hacia atrás, distribuyendo una porción de dicho

error a todas las neuronas de la capa anterior que aportaron para la obtención de la salida alcanzada. Este proceso se repite capa por capa hasta que todas las neuronas de la red hayan recibido una porción del error obtenido. El objetivo de realizar estos pasos, es crear una red cuyos pesos de cada neurona minimicen la función de costo, consiguiendo así un nivel de precisión inversamente proporcional al error alcanzado por cada entrada (mientras menos error, mayor sera la precisión alcanzada).

Matemáticamente, el descenso de gradiente esta basado en la observación de que si la función multivariable  $F(x)$  es definida y diferenciable en la vecindad del punto  $a$ , entonces  $F(x)$  decrece rápidamente si uno de ellos va desde  $a$  en dirección del gradiente negativo de  $F$  (donde la gradiente negativa esta representada por  $-\nabla F(a)$ ). Se deduce que si,  $a_{n+1} = a_n - \gamma \nabla F(a)$ , para un  $\gamma$  suficientemente pequeño, entonces  $F(a_n) \geq F(a_{n+1})$ . En otras palabras, el termino  $\gamma \nabla F(a)$  es substraído desde  $a$  porque se quiere mover a travez de la gradiente. Con esta observación en mente, se comienza con la suposición  $x_0$  para un mínimo local de  $F$ , y se considera la secuencia  $X_0, X_1, X_2, \dots$ , tal que:  $x_{n+1} = x_n - \gamma_n \nabla F(x_n)$ ,  $n \geq 0$ . Entonces nosotros tendremos  $F(x_0) \geq F(x_1) \geq F(x_2) \geq \dots$  esperando que la secuencia converga en el mínimo local esperado. Se debe tener en cuenta que el valor del tamaño del paso  $\gamma$  puede cambiar en cada iteración, obteniendo así la siguiente ecuación.

$$\gamma_n = \frac{(x_n - x_{n-1})^T [\nabla F(x_n) - \nabla F(x_{n-1})]}{\| \nabla F(x_n) - \nabla F(x_{n-1}) \|^2} \quad (2.7)$$

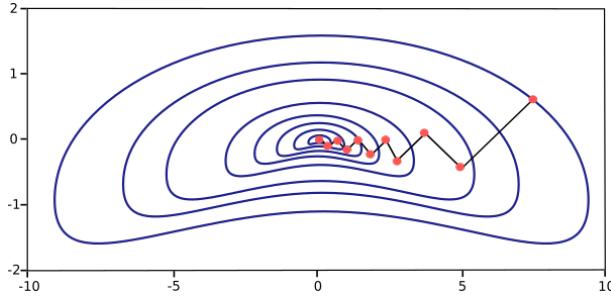


Figura 2.11: Descenso de gradiente.

Fuente: *5 algorithms to train a neural network, Neural Designer*.

## 2.7. Deep Learning

El *Deep Learning* es un concepto muy amplio, lo que implica que este tenga muchas definiciones. Sin embargo, de una forma muy general, se puede decir que el *deep learning* es un concepto que surge de la idea de imitar el cerebro humano por medio de una abstracción mas profunda de las redes neuronales, con el objetivo de crear una inteligencia que más se asemeje a la inteligencia humana, esta enfoque utiliza una capacidad de abstracción jerárquica, es decir, una representación de los datos de entrada en varios "niveles". A diferencia de las Redes Neuronales tradicionales, el *deep learning* utiliza multiples capas ocultas para la selección de características que son útiles para un mejor aprendizaje; de esta manera, la capa mas profunda

dentro de las capas ocultas, obtendrán la representación de características con mayor nivel de abstracción (reconocimiento de líneas, puntos, curvas y otros).

Este enfoque no es más que un conjunto de algoritmos de *machine learning*, que intenta detectar abstracciones de un alto nivel en datos, haciendo uso de arquitecturas compuestas de transformaciones no-lineales múltiples [3]. Existen varios tipos de aprendizajes, las cuales son tomadas como base para el entrenamiento de una red neuronal en general. Sin embargo, dos principales categorías son resaltadas por las arquitecturas profundas (como se muestra en la figura 2.12):

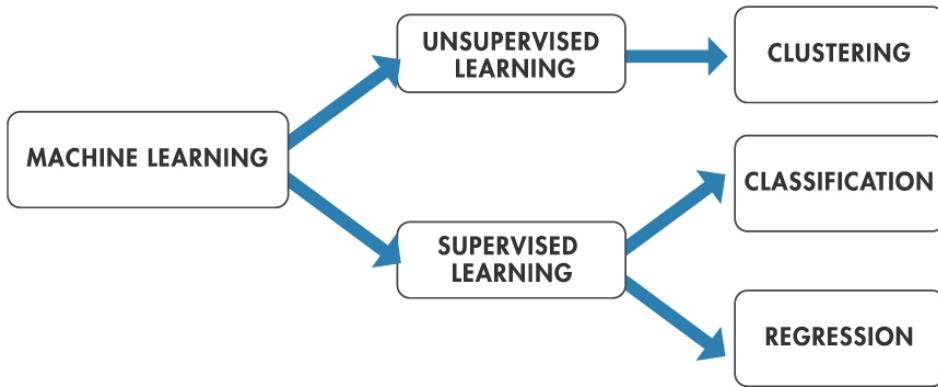


Figura 2.12: Tipos de aprendizaje.

Fuente: *Big Data with MATLAB*, Math Works.

- **Supervisado:** Se caracteriza porque su entrenamiento es controlado por un agente externo. Es decir, este agente externo guía el entrenamiento de la red mediante una comparación entre la salida esperada y la salida obtenida por medio de la red. En otras palabras, para realizar este proceso, la base de datos a entrenar necesita contener un campo que indique la salida que se espera por cada dato de entrada. Por ejemplo, sea una arquitectura dedicada a la reconocimiento y clasificación de números en imágenes, entonces, cada vez que pasemos una imagen como entrada a la red, esta imagen deberá ir acompañada de una etiqueta que indique cual es el número que se está pasando [26].

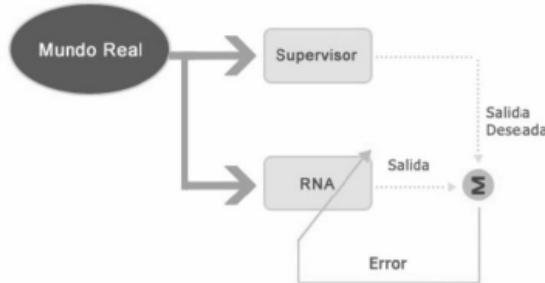


Figura 2.13: Aprendizaje supervisado.

Fuente: López S, Jesús A. Caicedo B, Eduardo F.

- **No supervisado:** Este enfoque sugiere que el aprendizaje sea realizado presentándole a la red los datos directamente, es decir, ahora no existe un agente externo, campo o etiqueta que indique a la red cuales son los datos que se le proporciona como entrada. La red aprende de ellos, agrupandolos de acuerdo a la similaridad de sus características, en algunos casos se realiza este tipo de agrupación de forma probabilística [26].

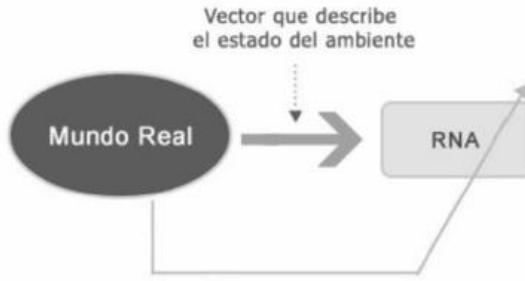


Figura 2.14: Aprendizaje no supervisado.

Fuente: López S, Jesús A. Caicedo B, Eduardo F.

- **Híbrido:** Algunas redes tienden a utilizar ambos tipos de aprendizaje para su fase de entrenamiento, ya sea comenzando por un pre-entrenamiento supervisado y finalizando con uno no supervisado o viceversa. Este enfoque se sigue con el objetivo de lograr un mejor ajuste, disminuyendo el tiempo de convergencia y entre otras funcionalidades [26].

## 2.8. Redes mas Comunes Consideradas dentro del Deep Learning

### 2.8.1. Autoencoder

Es un tipo de red neuronal artificial utilizada para el aprendizaje no supervisado de codificaciones eficientes. El objetivo de una autoencoder es aprender una representación (codificación) para un conjunto de datos, típicamente con el propósito de reducción de dimensionalidad. Recientemente, el concepto autoencoder se ha vuelto más ampliamente utilizado para el aprendizaje de modelos generativos de datos [34].

Un auto-codificador, o autoencoder, aprende a producir a la salida exactamente la misma información que recibe a la entrada. Por eso, las capas de entrada y salida siempre deben tener el mismo número de neuronas. Por ejemplo, como se muestra en la figura 2.15, si la capa de entrada recibe los píxeles de una imagen, se espera que la red aprenda a producir en su capa de salida exactamente la misma imagen que ha sido introducido [26].

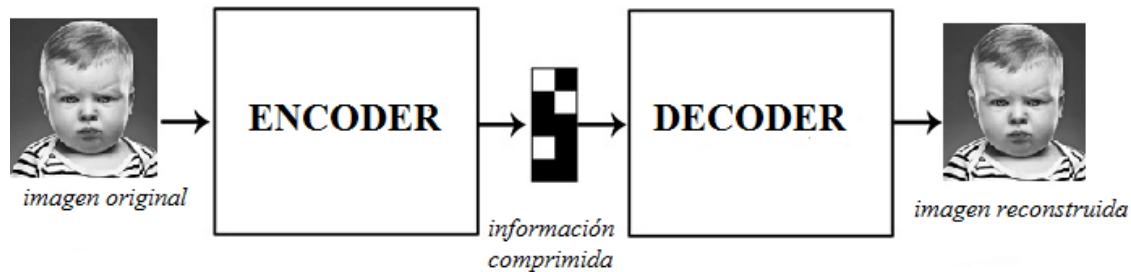


Figura 2.15: Arquitectura de una red neuronal Auto-encoder.

Fuente: Propio.

## 2.8.2. Redes Neuronales Recurrentes

Las Redes de Neuronas Recurrentes (*Recurrent Neural Networks*) no tienen una estructura de capas definida, sino que permiten conexiones arbitrarias entre todas las neuronas, incluso creando ciclos. Esto permite incorporar a la red el concepto de temporalidad, y permite que la red tenga memoria, porque los números que introducimos en un momento dado en las neuronas de entrada son transformados, y continúan circulando por la red incluso después de cambiar los números de entrada por otros diferentes [26].

Este tipo de redes, debido a la capacidad de retener información por cortos períodos de tiempo, es altamente utilizado en trabajos relacionados con el procesamiento y análisis de videos, donde las entradas a la red son un conjunto de frames (imágenes), siendo cada imagen la continuación de la anterior. Sin embargo, similar a las otras redes neuronales, la complejidad en términos de tiempo de ejecución (entrenamiento de la red) sigue siendo una limitación para experimentar con dichos trabajos, ya que el número de parámetros a optimizar siguen siendo miles de miles, debido a las múltiples conexiones que se presentan entre neuronas.

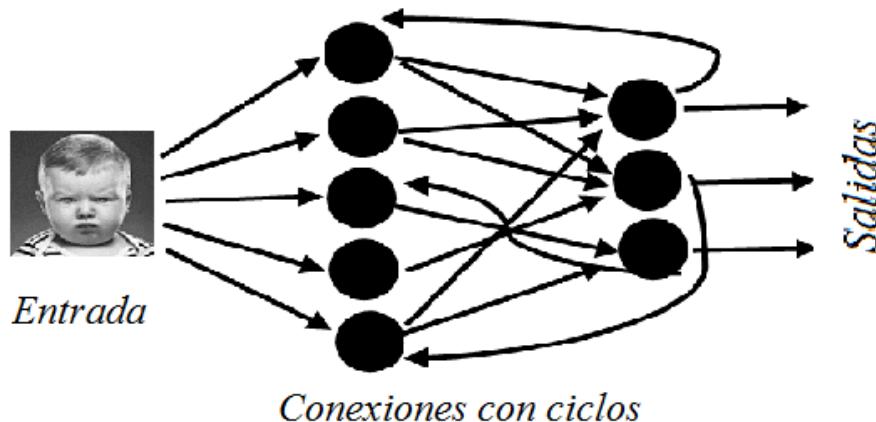


Figura 2.16: Arquitectura de una red neuronal Recurrente.

Fuente: Propio.

El diagrama 2.17 muestra una red neuronal recurrente siendo desarrollada. Donde  $x_t$  es

la entrada en el paso  $t$ ,  $s_t$  es el estado oculto en el paso del tiempo  $t$ , esta es la memoria de la red.  $s_t$  es calculada basada en los estados ocultos previos y la entrada al actual paso:  $s_t = f(Ux_t + Ws_{t-1})$ . La función  $f$  normalmente es una no linearidad. Finalmente,  $o_t$  es la salida del paso  $t$ , donde  $o_t = \text{softmax}(Vs_t)$ .

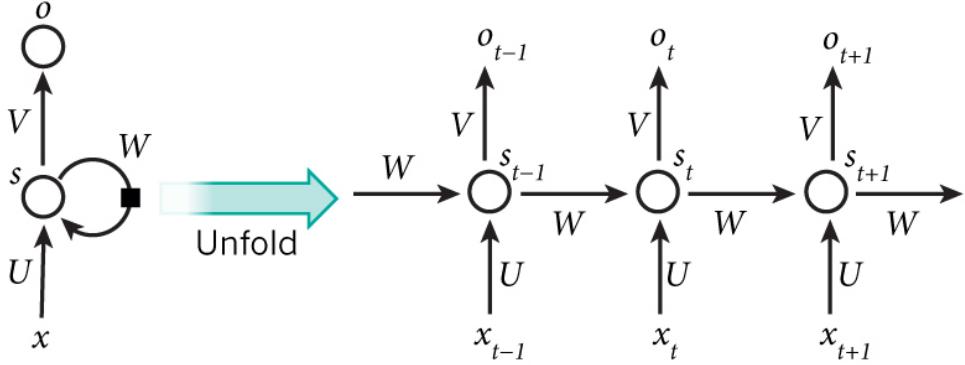


Figura 2.17: Red neuronal Recurrente.

Fuente: *Nature*.

### 2.8.3. Redes Neuronales Convolucionales

Las Redes Neuronales Convolucionales (*Convolution Neural Network*) a diferencia de las Redes Neuronales Recurrentes, mantienen el concepto de capas, pero cada neurona de una capa no recibe conexiones entrantes de todas las neuronas de la capa anterior, sino sólo de algunas. Esto favorece que una neurona se especialice en una región de la lista de números de la capa anterior, y reduce drásticamente el número de pesos y de multiplicaciones necesarias. Lo habitual es que dos neuronas consecutivas de una capa intermedia se especialicen en regiones solapadas de la capa anterior [25].

Muy diferente del esquema tradicional de las redes neuronales artificiales, estas redes presentan una gran variedad de tipos de capas, donde cada una de ellas tiene una función específica y esta puede repetirse más de una vez. La siguiente sección sera dedicada a describir con mayor detalles la arquitectura modelo de una Red Neuronal Convolutacional.

## 2.9. Arquitectura de una Red Neuronal Convolucional

Las Redes Neuronales Convolucionales son una estructura compuesta de varias fases entrenables, aprendiendo de cada una de las características con diferentes grados de abstracción. La entrada y salida de cada una de estas etapas son conjunto de arreglos llamados mapas de características, a la salida cada mapa de características representa una característica particular extraída de la imagen de entrada. Cada fase generalmente está compuesta por tres capas: Convolución, función no lineal y una capa de sub-muestreo. Las últimas capas por lo general son un conjunto de capas totalmente conectadas.

Una típica arquitectura de red neuronal convolucional para clasificación supervisada está basada en varias etapas (los tres tipos de capas mencionadas en el párrafo anterior: con-

volucion y sub-muestreo) seguidas de un clasificador (el conjunto de capas totalmente conectadas). Una de las primeras arquitecturas, es la red propuesta por Yann LeCun (figura 2.18). Esta red fue utilizada para resolver el problema de reconocimiento de caracteres manuscritos, utilizando una arquitectura con dos fases.

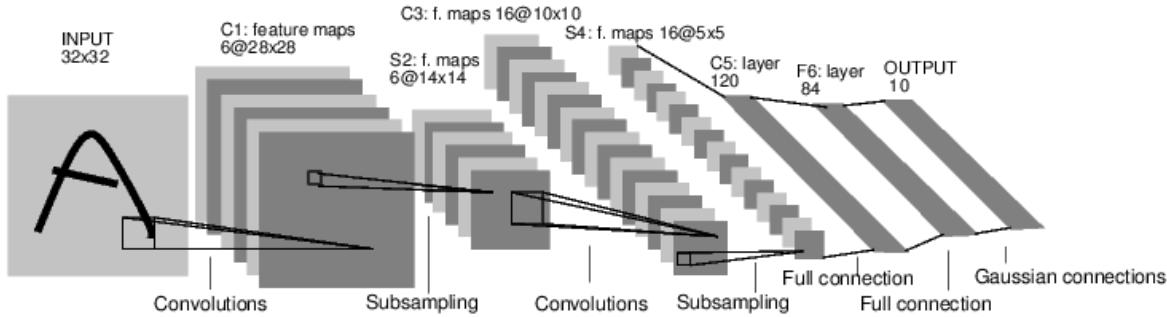


Figura 2.18: Arquitectura de una red neuronal Convolucional.

Fuente: Source: Yann LeCun, 1998.

Esta red denominada Let-Net, toma una imagen como entrada, la capa siguiente esta constituido por un conjunto de filtros de convolución (mapas de características), seguida por una capa de sub-muestreo encargada de la reducción de dimensionalidad, seleccionando así solo los datos mas relevantes. La capa de convolución está compuesta por seis mapas de características, donde cada uno de ellos no es mas que una matriz con valores asociados. A continuación, se encuentra la capa de sub-muestreo, que, como se menciono antes, agrupa las salidas de las nuevas imágenes originadas por la capa de convolución, generando así la misma imagen pero con dimensiones menores e información mas relevante. Este nuevo mapa de características sirve de entrada para la siguiente fase dedicada a encontrar características de mayor abstracción. Una cosa importante que cabe resaltar, es que, a medida que se avanza en las fases se aprenden características mas relevantes, pero mas invariantes a posición (por el sub-muestreo). Finalmente, las capas totalmente conectadas se encargan de evaluar las posibles combinaciones de las características aprendidas para lograr clasificar las imágenes dadas [25].

### 2.9.1. Capa de Convolución

La capa de convolución es el bloque de construcción básico de una red, esta capa hace la mayor parte del trabajo pesado computacional, debido a que se realizan multiplicaciones de matrices entre la imagen y los filtros de convolución.

- **Visión general e intuición sin cerebro.** Los parámetros de la capa de convolución consisten en un conjunto de filtros que durante la fase de entrenamiento son aprendidos. Cada filtro es pequeño espacialmente (a lo largo de la anchura y altura), este filtro es aplicado a través de toda la imagen de entrada. Por ejemplo, un filtro típico en una primera capa de una red neuronal convolucional podría tener un tamaño de 5x5x3 (es decir, 5 píxeles anchura y la altura, y 3 ya que las imágenes tienen profundidad 3, los

canales de color). Durante la operación de convolución entre el filtro y la imagen de entrada, éste se desliza a través del ancho y la altura del volumen de entrada y calcula el productos escalares entre estos dos elementos. A medida que se desplaza el filtro sobre la anchura y la altura del volumen de entrada se produce un mapa de activación de 2 dimensiones que da las respuestas de ese filtro en cada posición espacial. Intuitivamente, la red aprenderá filtros que se activan cuando ven algún tipo de función visual, como un borde en una determinada orientación o una mancha de un cierto color. Después se tendrá todo un conjunto de filtros en cada capa de convolución, y cada uno de ellos va a producir un mapa de activación de 2 dimensiones por separado. Finalmente se apilan estos mapas de activación a lo largo de la dimensión de la profundidad y producen el volumen de salida. [6].

- **La vista del cerebro.** Cada entrada en el volumen de salida 3D también se puede interpretar como una salida de una neurona que mira sólo una pequeña región en los parámetros de entrada y comparte con todas las neuronas de la izquierda y derecha (ya que todos estos números resultaría de aplicar el mismo filtro). [6].
- **Conectividad local.** Cuando se trata de entradas de alta dimensión como las imágenes, como se vio anteriormente, no es práctico conectar neuronas a todas las neuronas en la capa anterior. En su lugar, se va a conectar cada neurona a sólo una región local del volumen de entrada. La extensión espacial de esta conectividad es un hiperparámetro llamado campo receptivo de la neurona (equivalente al tamaño del filtro). La extensión de la conectividad a lo largo del eje de profundidad es siempre igual a la profundidad del volumen de entrada. Es importante destacar nuevamente esta asimetría en cómo tratamos las dimensiones espaciales (anchura y altura) y la dimensión de la profundidad: Las conexiones son locales en el espacio (a lo largo del ancho y la altura), pero siempre llenas a lo largo de toda la profundidad del volumen de entrada [6].

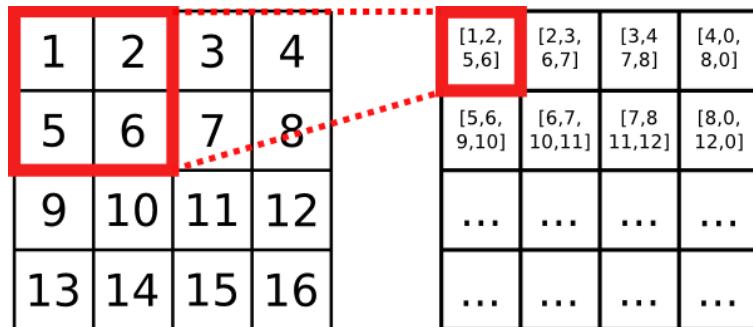


Figura 2.19: Ejemplo de convolución con un filtro de dimensiones de  $2 \times 2$ .

Fuente: *¿Qué es y como funciona Deep Learning?*, Rubén López.

- **Pseudo – Código.** La representación de la convolución por medio de una formula matemática se muestra en la ecuación 2.8. Los valores de un pixel en la imagen de salida se calculan multiplicando cada valor del kernel por los valores de píxeles de la

imagen de entrada correspondientes. Esto se puede describir algorítmicamente con el siguiente pseudo-código:

$$V = \frac{\sum_{i=0}^{q1} \sum_{j=0}^{q2} f_{i,j} * k_{i,j}}{F} \quad (2.8)$$

---

**Algorithm 1** Pseudo-Código Convolución

La convolución de una image  $f(x,y)$  con un kernel  $k(x,y)$  con dimensiones  $h \times w$  y  $(2h+1) \times (2w+1)$  respectivamente produce una nueva imagen  $g(x,y)$

---

```

procedure CONVOLUCIÓN( $f, k$ )            $\triangleright$  Convolución de la imagen f con el kernel k
    for  $y := 1$  to  $W$  do
        for  $x := 1$  to  $H$  do
             $sum = 0$ 
            for  $i := -h$  to  $h$  do
                for  $j := -w$  to  $w$  do
                     $sum = sum + k(j, i) * f(x - j, y - i)$ 
             $g(x, y) = sum$ 
    return  $g$                                  $\triangleright$  Resultado de la convolución entre f y k

```

---

Donde:

- $f_{i,j}$ : Corresponde al pixel en la posición  $i, j$  de la imagen  $f$  respecto al kernel  $k$ .
- $k_{i,j}$ : Corresponde al pixel en la posición  $i, j$  del kernel  $k$ .
- $q1 \times q2 = (2h + 1) \times (2w + 1)$ : Representa las dimensiones del kernel.
- $F$ : Es la suma de los coeficientes del kernel (1 si la suma es igual a 0).
- $g(i, j)$ : Representa el valor de salida del pixel en la posición  $i, j$ .

### 2.9.2. Submuestreo

La arquitectura tradicional de una red neuronal convolucional sugiere que se inserte una capa de sub-muestreo después de cada capa de convolución. Su función es reducir progresivamente el tamaño espacial de la imagen de entrada con el objetivo de reducir la cantidad de parámetros a ser calculados en la red, y por lo tanto, también para controlar el sobre ajuste. La capa de agrupación funciona independientemente en cada segmento de profundidad de la entrada y la redimensiona espacialmente, dependiendo de las dimensiones de la ventana de sub-muestreo. Existen diferentes funciones de agrupación para realizar este tipo de operación, funciones como: MAX, MIN, MODA, MEDIANA, etc, cada una de estas funciones se desenvuelve de manera muy diferente, dependiendo del problema que ese pretende resolver. La forma más común es una capa de agrupación con filtros de tamaño 2x2 y función de agrupación MAX, aplicando un salto de dos pixeles hacia abajo y hacia arriba para realizar el desplazamiento e iterar la operación de sub-muestreo sobre toda la imagen, de esta forma

descartando el 75 % de las activaciones. En este caso, cada operación MAX tomaría un máximo de 4 números (pequeña región de dimensiones de 2x2). La dimensión de profundidad no cambia. Más generalmente, la capa de agrupación tiene estas principales características:

- Acepta un volumen de tamaño  $W1 \times H1 \times D1$
- Requiere 2 hiperparámetros
  - Su extensión espacial  $F$
  - El desplazamiento  $S$
- Produce un volumen de tamaño:  $W2 \times H2 \times D2$ , donde :
  - $W2 = \frac{W1-F}{S+1}$
  - $H2 = \frac{H1-F}{S+1}$
  - $D2 = D1$
- Introduce parámetros cero, ya que calcula una función fija de la entrada.
- Tener en cuenta que no es común utilizar cero como relleno(*padding*) para agrupar capas.

En la práctica solo fueron encontradas dos variaciones comunes de la capa de sub-muestreo con función de agrupación MAX: Una capa de agrupación con  $F = 3$ ,  $S = 2F$ ,  $S = 2$  (también llamada superposición de agrupación) y más comúnmente  $F = 2$ ,  $S = 2F$ ,  $S = 2$ . Los tamaños de agrupación con campos receptivos más grandes son demasiado destructivos [6].

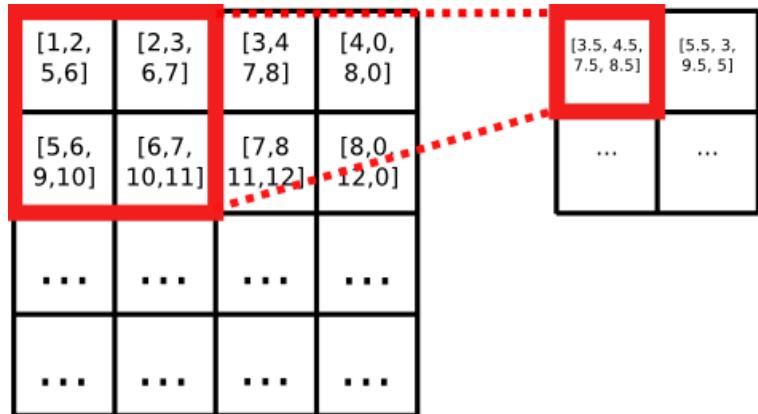


Figura 2.20: Ejemplo de Submuestreo con una ventana de 2X2 y calculando el promedio

Fuente: *¿Qué es y como funciona Deep Learning?*, Rubén López.

### 2.9.3. Capa de normalización

Existen muchos tipos de capas de normalización propuestas hasta la actualidad, algunos de ellos fueron creados con la intención de asemejar mas la red al comportamiento real del cerebro. Sin embargo, estas capas a pesar de aparentar ser un gran aporte para conseguir una inteligencia artificial pura, han caido debido a que en la práctica su aporte es casi nula.

El objetivo de una capa de normalización es, como su mismo nombre lo dice, normalizar las activaciones de la capa anterior, es decir, se aplica una transformación que mantiene la activación de cierre media en 0 y la desviación estándar cerca de 1 [6].

### 2.9.4. Capa totalmente conectada

Las neuronas en una capa completamente conectada tienen conexiones completas con todas las activaciones (neuronas) en la capa anterior, haciendo de este tipo de capas un grafo muy denso o grafo completo (figura 2.21), el cual computacionalmente es muy costoso, sin embargo, generalmente dentro de las redes neuronales convolucionales, a lo mucho son usadas 2 a 3 capas totalmente conectadas, haciendo de este un trabajo ligeramente mas facil de computar. Este tipo de capas son como las que se ven en las redes neuronales regulares. Por tanto, sus activaciones pueden calcularse con una multiplicación matricial. [6].

Esta capa basicamente toma un volumen como entrada y su salida es un vector de N dimensiones, donde N es el número de clases correspondientes al problema que se esta resolviendo (en el caso de clasificación). Por ejemplo, si se esta trabajando en una red, la cual sera capaz de reconocer dígitos a partir de imágenes como entrada, entonces, el número de salidas sera 10 (una salida para cada números del 0 a 9). Cada número en este vector de N dimensiones representa la probabilidad que este tiene para pertenecer a una determinada clase (esta probabilidad es obtenida gracias a la función de normalización, la cual se describe en la sub-sección 2.9.5), donde la suma de todas las probabilidades es 1. Sea el problema antes planteado, si el vector resultante es  $[0, 0, 1, 0, 0, 5, 0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 8]$ , la probabilidad de que la imagen sea un 1 es de 10 %, mientras que la probabilidad de que sea un 9 es de 80 %.

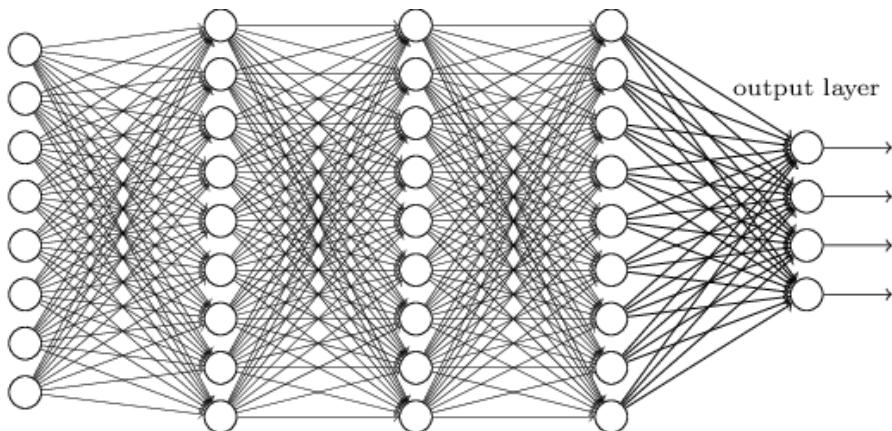


Figura 2.21: Capa totalmente conectada

Fuente: Michael A. Nielsen, <http://neuralnetworksanddeeplearning.com/chap6.html>

## 2.9.5. Función de normalización(Softmax)

La función de normalización softmax, también conocida como función exponencial normalizada, no es mas que un nombre para la regresión lineal multinomial o simplemente clase múltiple de regresión logística. En su esencia, regresión de softmax es una generalización de la regresión logística que podemos utilizar para la clasificación de clase múltiple (bajo el supuesto de que las clases son mutuamente excluyentes). En cambio, utilizamos el modelo de regresión logística (estándar) en tareas de clasificación binario.

Esta función tiene como entrada el vector de N dimensiones obtenida por la capa totalmente conectada, y como salida, un vector de probabilidades. La figura 2.22 muestra como esta función se integra en la red.

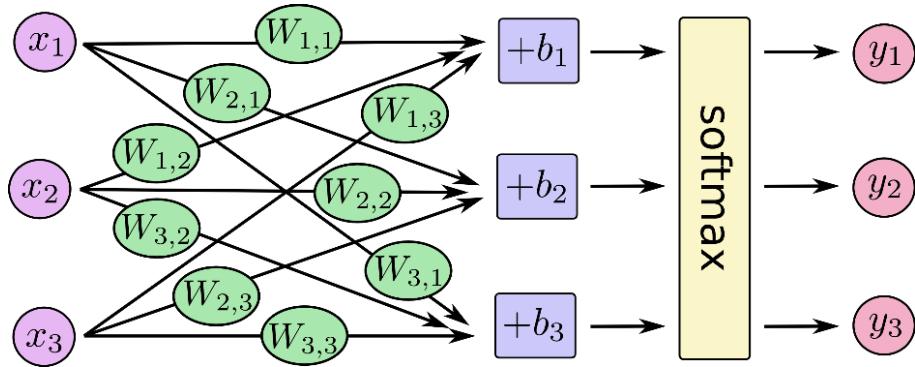


Figura 2.22: Arquitectura de una CNN con Softmax

Fuente: Samuel Salvatella, <http://ssalva.bitballoon.com/blog/2016-08-30-tensorflow/>

En su representación matematica, como se menciono antes, esta función es una generalización de la función logística que permite la utilización de un vector de dimensión. La función está dada por la ecuación 2.9.

$$P(Y = j|Z^i) = \phi_{softmax}(Z^i) = \frac{\exp^{Z^i}}{\sum_{j=0}^k \exp^{Z^i_j}} \quad (2.9)$$

Donde:

$$Z = w_0x_0 + w_1x_1 + \dots + w_mx_m = \sum_{l=0}^m w_lx_l = w^T x$$

## 2.10. Entrenamiento de una Red Convolucional

El proceso de entrenamiento en un red neuronal convoluciona, se basa en el algoritmo *BackPropagation* (como se describió en la subsection 2.6), el cual consiste en calcular una función objetivo que minimiza el error, por medio de la retro propagación de este, hacia las capas anteriores, de tal manera que los pesos de las conexiones entre neuronas son ajustadas.

El algoritmo *BackPropagation*, como se describio anteriormente, es muy intuitivo. Este sigue la siguiente secuencia de pasos:

---

**Algorithm 2** Pseudo-Código softmax

---

```
procedure SOFTMAX(neuron_output)
    sum = 0
    for v in neuron_output do
        sum = sum + v
    sum = math.exp(sum)
    proba[]
    for i in range(size(neuron_output)) do
        proba[i] = math.exp(neuron_output[i])/sum
    return proba
```

---

- Primero, se proporciona datos de entrada a la red.
- Segundo, son propagadas dichas entradas hasta la capa de salida, con pesos iniciales definidos o aleatorios.
- Tercero, se calcula el error en la capa de salida utilizando la función de costo que se muestra en la ecuación 2.10, donde *target* se refiere a la salida esperada y *output* a la salida obtenida por la red.

$$E_{total} = \sum \frac{1}{2}(target - output)^2 \quad (2.10)$$

- Cuarto, se propaga dicho error hacia las neuronas ocultas (hacia atrás) utilizando el algoritmo *backpropagation*.
- Quinto, se actualizan los pesos de las conexiones.

## 2.11. Sobre las Expresiones Faciales

### 2.11.1. Paul Ekman

Paul Ekman es un psicólogo que tiene como foco de investigación el estudio de las emociones y las expresiones faciales que generan estas [9]. Él comenzó su investigación en la comunicación no verbal en la década de 1950, el desarrollo de maneras sistemáticas para medir el lenguaje corporal. En el proceso, descubrió que, a través de la investigación empírica, pudo identificar constantemente las expresiones faciales creadas por el movimiento de los músculos de la cara. También introdujo la detección de microexpresiones que mostró que pueden ser útiles en la detección de mentiras con un nivel de confianza [10]. Además, desarrolló el Sistema de Codificación Facial de Acciones (en inglés "Facial Action Coding System", FACS) [11].

## 2.11.2. Las seis emociones básicas

Antes de Ekman se introdujera en esta linea de investigación (reconocimiento de emociones y sus expresiones faciales), se creía ampliamente (por antropólogos incluyendo Margaret Mead) que las expresiones faciales y las emociones que ellos representan se determinaron por la cultura – que las personas aprendieron a hacer y leer las expresiones faciales de sus sociedades. Ekman se dispuso a probar esta idea en 1968. Él viajó a Papúa Nueva Guinea para estudiar las expresiones faciales de los miembros de la tribu Fore apartada, donde aprendió que podían identificar constantemente las emociones en las expresiones faciales por mirar fotos de la gente de otras culturas, a pesar de que la tribu no había sido expuesta a cualquier otra cultura exterior. Pudiendo así evidenciar que las expresiones faciales son interculturales, su investigación reveló que existe un conjunto universal de ciertas expresiones faciales se utilizan tanto en el mundo occidental y oriental, sin embargo existe un grupo de científicos a nivel mundial que están en desacuerdo con estos resultados hechos por Paul Ekman. Esta lista de expresiones faciales universales, que Ekman publicó en el año 1972, dispone de las seis emociones básicas [11].

### ■ Cólera:

- **Descripción.-** El antagonismo hacia una persona o un objeto a menudo se sentía después de que uno siente que ha sido agraviado u ofendido.
- **Movimientos musculares faciales.-** La reducción de las cejas, apretar y estrechar los labios, los ojos mirando, apretando los párpados inferiores, con menos frecuencia, empujando la mandíbula hacia adelante.



Figura 2.23: Expresión Facial de Cólera

Fuente: *Las 6 emociones básicas*, Paul Ekman

### ■ Felicidad:

- **Descripción.-** Agradable sensación de satisfacción y bienestar.
- **Movimientos musculares faciales.-** Smiling – tirando hacia arriba comisuras de la boca, contrayendo los músculos grandes orbitales alrededor de los ojos.



Figura 2.24: Expresión Facial de Felicidad

Fuente: *Las 6 emociones básicas*, Paul Ekman

■ Sorpresa:

- **Descripción.-** Sensación de malestar o sorpresa ante un hecho inesperado.
- **Movimientos musculares faciales.-** Levantando las cejas altas (que puede causar arrugas en la frente), abriendo los ojos como platos, dejando caer la mandíbula tan boca es ágate.



Figura 2.25: Expresión Facial de Sorpresa

Fuente: *Las 6 emociones básicas*, Paul Ekman

■ Asco:

- **Descripción.-** Desagrado intenso o condena causada por algo ofensivo o repulsiva.
- **Movimientos musculares faciales.-** La reducción de las cejas, curvando el labio superior, arrugando la nariz.



Figura 2.26: Expresión Facial de Asco

Fuente: *Las 6 emociones básicas*, Paul Ekman

■ Tristeza:

- **Descripción.-** Sentimiento de infelicidad o tristeza.
- **Movimientos musculares faciales.-** Los párpados caídos, la reducción de las esquinas de la boca, labios fruncidos, los ojos bajos.



Figura 2.27: Expresión Facial de Tristeza

Fuente: *Las 6 emociones básicas*, Paul Ekman

■ Miedo:

- **Descripción.-** Sensación de aprehensión provocada por la percepción de peligro, amenaza o imposición de dolor.
- **Movimientos musculares faciales.-** Levantando las cejas / dibujar las cejas juntas, tensando los párpados inferiores, que se extiende horizontalmente labios, la boca ligeramente abierta.



Figura 2.28: Expresión Facial de Miedo

Fuente: *Las 6 emociones básicas*, Paul Ekman

### 2.11.3. Otras expresiones faciales

Los hallazgos de Ekman sobre las expresiones faciales universales revelaron el carácter intercultural de la relación entre la comunicación no verbal y la emoción, sin embargo, las teorías de Ekman han evolucionado desde que ideó su lista de emociones básicas. En la década de 1990, añadió una serie de otros a la lista de emociones universales, aunque hizo hincapié en que no todos ellos pueden ser identificados utilizando expresiones faciales. Estas emociones adicionales son [11].

- Diversión
- Desprecio
- Contentamiento
- Vergüenza
- Emoción
- Culpa
- El orgullo de los logros
- Alivio
- Satisfacción
- Placer sensorial
- Neutro (sin emociones)

# **Parte III**

## **Desarrollo del Proyecto**

# Capítulo 3

## Desarrollo del Detector de Rostros y la Arquitectura de Red Neuronal Convolucional

### 3.1. Detección de Rostros

En este trabajo se optó por utilizar como etapa inicial dentro de la fase de consultas a la red, la construcción de un detector de rostros, debido a que, al momento de desarrollar una aplicación orientada a las necesidades del mundo real, este no solo recibira como entrada imágenes que contengan exactamente el rostro de la persona, sino, imágenes con el cuerpo completo o algunas partes adicionales aparte del rostro. Sin embargo, en objetivo del trabajo es poder detectar la expresión facial de una persona, para lo cual, basta con tener como entrada a la red una imagen que delimita el rostro de la persona. De ahí, la necesidad de utilizar un algoritmo de detección de rostros para la extracción de la región de interés que posteriormente servira como entrada para la red neuronal convolucional encargada de reconocer la expresión facial correspondiente. La figura 3.1 muestra un ejemplo de una imagen de entrada, en la cual, se puede observar detalles adicionales aparte del rostro (el sombrero y el fondo), los cuales no aportan características relevantes que ayuden al reconocimiento de la expresión facial.

Para esta etapa se utilizó el detector de objetos *haar cascade*, un algoritmo muy utilizado, cuya implementación puede ser encontrado en distintas librerías orientadas al procesamiento de imágenes, tales como OpenCV<sup>5</sup>. Como se describió en la sección 2.2.1, este algoritmo utiliza técnicas de *machine learning*. Su proceso de entrenamiento se realiza con imágenes positivas y negativas (imágenes que representan y no representan rostros), creando así un modelo capaz de detectar rostros, basándose en la detección de características *haar*. La entrada para esta etapa es una imagen cualquiera, el proceso consiste en detectar el rostro en dicha imagen (en caso exista algun rostro) y extraerlo en otra imagen en escala de grises, la cual tendrá un tamaño aproximado de 48x48 píxeles (dependiendo de las dimensiones del rostro). Esta

---

<sup>5</sup>OpenCV es una librería *open source* que contiene algoritmos relacionados con el área de visión por computador, <http://opencv.org/>

última imagen será la entrada para el modelo en la fase de consultas. Nótese que para la detección de un rostro y la asignación de su respectiva expresión facial, no es necesario mantener la imagen a color, puesto que este no es una característica necesaria para conseguir el objetivo. La figura 3.2 muestra los pasos a seguir para la detección y extracción del rostro en una imagen.



Figura 3.1: Ejemplo de una imagen de entrada.

Fuente: Consuelo Ferrús, <http://www.acompasando.org/orar-el-asombro/>

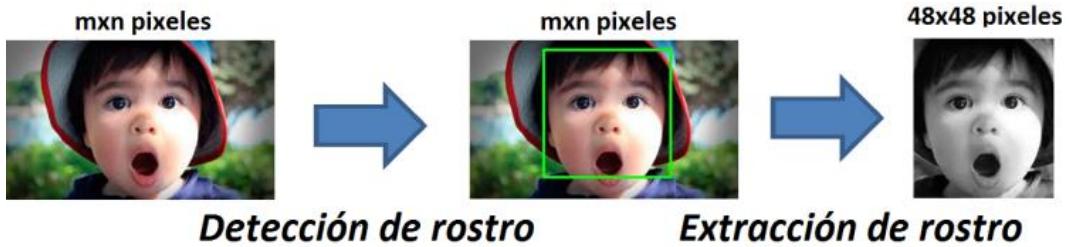


Figura 3.2: Proceso de detección de rostro.

Fuente: Propio

Se optó por la utilización del detector de rostros *haar cascade*, debido a que este es ampliamente utilizado por el nivel de precisión que posee [27]. Sin embargo, esta técnica aun presenta algunas fallas cuando el rostro presenta algún tipo de oclusión (figura 3.3). Este tipo de problema también puede ser solucionado construyendo una red convolucional orientada a la detección o localización de rostros, o por medio de otro tipo de técnicas tradicionales basadas en la extracción de características. Debido al tiempo y bajos recursos computacionales, se optó por proponer este tipo de enfoques como trabajos futuros.



Figura 3.3: Ejemplo de rostros con oclusión.

Fuente: Face Dataset, Universidad Politécnica de Catalunya.

## 3.2. Experimentación en la Elección de Parámetros y Capas en la Construcción de la Arquitectura CNN

La creación de un modelo robusto a partir de la utilización de redes neuronales convolucionales, es una tarea muy complicada, debido a que en la actualidad no existen estudios que indique cual es la configuración correcta que esta debe seguir. Es así, que todos los trabajos que utilizan cualquier técnica perteneciente a *deep learning*, se basan en la experimentación sobre diversar configuraciones en la arquitectura, estas configuraciones estan relacionadas con el número de capas de convolución, sub-muestreo, totalmente conectadas, tipos de funciones de activacion y normalización. Otros elementos muy importantes que tambien son considerados al momento de realizar experimentos, son los parámetros de cada capa, tales como: el número y tamaño de los filtros en la capa de convolución, operaciones de agrupación en la capa de sub-muestreo, etc.

Sin embargo, entrenar un red neuronal no es una tarea fácil, debido a que se tienen que optimizar miles de parámetros para la creación de un buen modelo, asi como las miles de multiplicaciones de matrices que tienen que llevarse a cabo para realizar la operación de convolución. Es así que el número de experimentos que se puedan realizar, depende mucho de la insfraestructura y *hardware* sobre el cual se esta trabajando, siendo este una gran limitante para una extensa experimentación.

Según lo explicado en los parrafos anteriores, para encontrar la correcta configuracion de una arquitectura basada en las redes neuronales convolucionales que resuelva el problema de reconocimiento de expresiones faciales, se realizaron muchos experimentos, cada uno con una configuracion diferente que la anterior. En total se evaluaron 3 arquitecturas con distintas configuracion de capas y parametros, en cada una de estas se calculo la precisión y error utilizando las bases de datos FER2013 y CK.

- Conv-Conv-Pool-Conv-Conv-Pool-FC

Base de Datos FER2013								
Conv	Conv	Pool	Conv	Conv	Pool	FC	Precisión	Error
64 (2x2)	64 (2x2)	2x2 (Max)	32 (4x4)	32 (4x4)	2x2 (Max)	1024x1024	45.18%	54.82%
32 (4x4)	32 (2x2)	3x3 (Avg)	32 (3x3)	64 (3x3)	3x3 (Min)	500x1024	40.72%	59.28%
16 (5x5)	32 (4x4)	4x4 (Min)	16 (2x2)	16 (2x2)	4x4 (Max)	1024x500	43.24%	56.76%
32 (4x4)	16 (4x4)	2x2 (Max)	64 (3x3)	32 (4x4)	2x2 (Avg)	1024x1024	30.1%	69.9%

Tabla 3.1: Evaluación de la arquitectura 1 y sus parámetros, FER2013

Base de Datos CK+								
Conv	Conv	Pool	Conv	Conv	Pool	FC	Precisión	Error
64 (2x2)	64 (2x2)	2x2 (Max)	32 (4x4)	32 (4x4)	2x2 (Max)	1024x1024	88.45%	11.55%
32 (4x4)	32 (2x2)	3x3 (Avg)	32 (3x3)	64 (3x3)	3x3 (Min)	500x1024	80.12%	19.88%
16 (5x5)	32 (4x4)	4x4 (Min)	16 (2x2)	16 (2x2)	4x4 (Max)	1024x500	84.24%	15.76%
32 (4x4)	16 (4x4)	2x2 (Max)	64 (3x3)	32 (4x4)	2x2 (Avg)	1024x1024	76.11%	23.89%

Tabla 3.2: Evaluación de la arquitectura 1 y sus parámetros, CK+

- Conv-Pool-Conv-Pool-FC Arquitectura Propuesta

Base de Datos FER2013						
Conv	Pool	Conv	Pool	FC	Precisión	Error
64 (2x2)	2x2 (Max)	16 (4x4)	2x2 (Max)	1024x1024	39.81%	60.19%
32 (4x4)	3x3 (Avg)	32 (3x3)	3x3 (Max)	500x1024	53.78%	46.22%
16 (5x5)	4x4 (Min)	16 (2x2)	4x4 (Max)	1024x500	27.14%	72.86%
<b>32 (4x4)</b>	<b>2x2 (Max)</b>	<b>64 (4x4)</b>	<b>2x2 (Max)</b>	<b>1024x1024</b>	<b>57%</b>	<b>43%</b>

Tabla 3.3: Evaluación de la arquitectura 2 y sus parámetros, FER2013

Base de Datos CK+						
Conv	Pool	Conv	Pool	FC	Precisión	Error
64 (2x2)	2x2 (Max)	16 (4x4)	2x2 (Max)	1024x1024	74.54%	25.46%
32 (4x4)	3x3 (Avg)	32 (3x3)	3x3 (Max)	500x1024	57.86%	42.14%
16 (5x5)	4x4 (Min)	16 (2x2)	4x4 (Max)	1024x500	74.54%	25.46%
<b>32 (4x4)</b>	<b>2x2 (Max)</b>	<b>64 (4x4)</b>	<b>2x2 (Max)</b>	<b>1024x1024</b>	<b>91%</b>	<b>9%</b>

Tabla 3.4: Evaluación de la arquitectura 2 y sus parámetros, CK+

- Conv-Pool-Pool-Conv-Conv-Pool-FC

Base de Datos FER2013								
Conv	Pool	Pool	Conv	Conv	Pool	FC	Precisión	Error
64 (2x2)	2x2 (Max)	2x2 (Max)	32 (4x4)	16 (4x4)	2x2 (Max)	2048x2048	35.17%	64.83%
<b>64 (4x4)</b>	<b>3x3 (Avg)</b>	<b>3x3 (Avg)</b>	<b>64 (3x3)</b>	<b>32 (3x3)</b>	<b>2x2 (Max)</b>	<b>500x1024</b>	<b>52.18%</b>	<b>47.82%</b>
32 (5x5)	4x4 (Min)	3x3 (Min)	16 (2x2)	16 (2x2)	4x4 (Max)	512x1024	30.70%	69.30%
16 (4x4)	2x2 (Max)	2x2 (Max)	64 (4x4)	64 (4x4)	3x3 (Max)	1024x1024	50.15%	49.85%

Tabla 3.5: Evaluación de la arquitectura 3 y sus parámetros, FER2013

Base de Datos CK+								
Conv	Pool	Pool	Conv	Conv	Pool	FC	Precisión	Error
64 (2x2)	2x2 (Max)	2x2 (Max)	32 (4x4)	16 (4x4)	2x2 (Max)	2048x2048	62.14%	37.86%
<b>64 (4x4)</b>	<b>3x3 (Avg)</b>	<b>3x3 (Avg)</b>	<b>64 (3x3)</b>	<b>32 (3x3)</b>	<b>2x2 (Max)</b>	<b>500x1024</b>	<b>80.26%</b>	<b>19.74%</b>
32 (5x5)	4x4 (Min)	3x3 (Min)	16 (2x2)	16 (2x2)	4x4 (Max)	512x1024	58.54%	41.46%
16 (4x4)	2x2 (Max)	2x2 (Max)	64 (4x4)	64 (4x4)	3x3 (Max)	1024x1024	78.12%	21.88%

Tabla 3.6: Evaluación de la arquitectura 3 y sus parámetros, CK+

De acuerdo con los experimentos realizados, se puede observar que la mejor configuración de capas es: convolución-pooling-convolución-pooling-FC, con 32 y 64 filtros de tamaño  $4 \times 4$  en la primera y segunda capa de convolución respectivamente, ventanas de agrupación de  $2 \times 2$  en las dos capas de sub-muestreo y un total de  $1024 \times 1024$  neuronas en las últimas capas totalmente conectadas.

Según los resultados mostrados en las tablas 3.1 y 3.2 se puede concluir que, la utilización de dos capas consecutivas de convolución es una mala elección, debido a que el hecho de aplicar dos filtros consecutivos hace que la información de la imagen se pierda más rápidamente que

lo normal, ya que el tamaño de la imagen se reduce significativamente después de cada una de estas operaciones. También, las tablas 3.5 y 3.6 muestran resultados muy por debajo de la segunda configuración (la mejor obtenida en los experimentos realizados), similar que la explicación anterior, se concluye que se dan estos resultados debido a la utilización de dos capas consecutivas de sub-muestreo, ya que el objetivo de estas capaz es reducir las dimensiones de la imagen solo preservando información relevante, sin embargo, al realizar una operación de sub-muestreo detrás de otra, hace que se pierda más información de la necesaria.

### 3.3. Arquitectura Propuesta

De acuerdo con los experimentados mostrados en la sección 3.10, la arquitectura que consiguió los mejores resultados sigue la siguiente configuración: la entrada consta de una imagen de 48x48 píxeles en escala de gris, que es el resultado de la fase de detección y extracción de rostro, seguido de una capa de 32 convoluciones con filtro de 4x4 sin solapamiento, luego se aplica un sub muestreo de 2x2 con función MAX<sup>6</sup>, seguido de una capa de 64 convoluciones con filtros de 2x2 sin solapamiento, para posteriormente aplicar un sub muestreo también de dimensiones 2x2 con función MAX, aplicada las convoluciones y sub muestreo se procede a aplicar el Dropout<sup>7</sup> con 20 %, seguido de dos capas, con 1024 neuronas totalmente conectadas cada una. Finalmente, para la clasificación se aplica la función de normalización Softmax, la cual toma 6 clases, las cuales representarán al número de expresiones a reconocer.

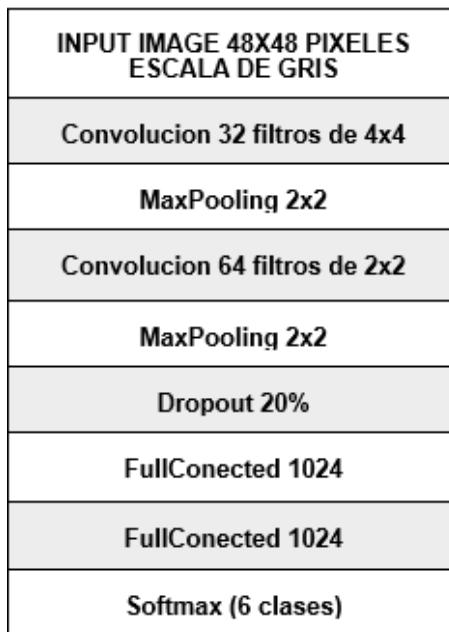


Tabla 3.7: Arquitectura del modelo propuesto

---

<sup>6</sup>Función que determina el máximo de n números.

<sup>7</sup>Una forma simple de prevenir el *overfitting* en redes neuronales.

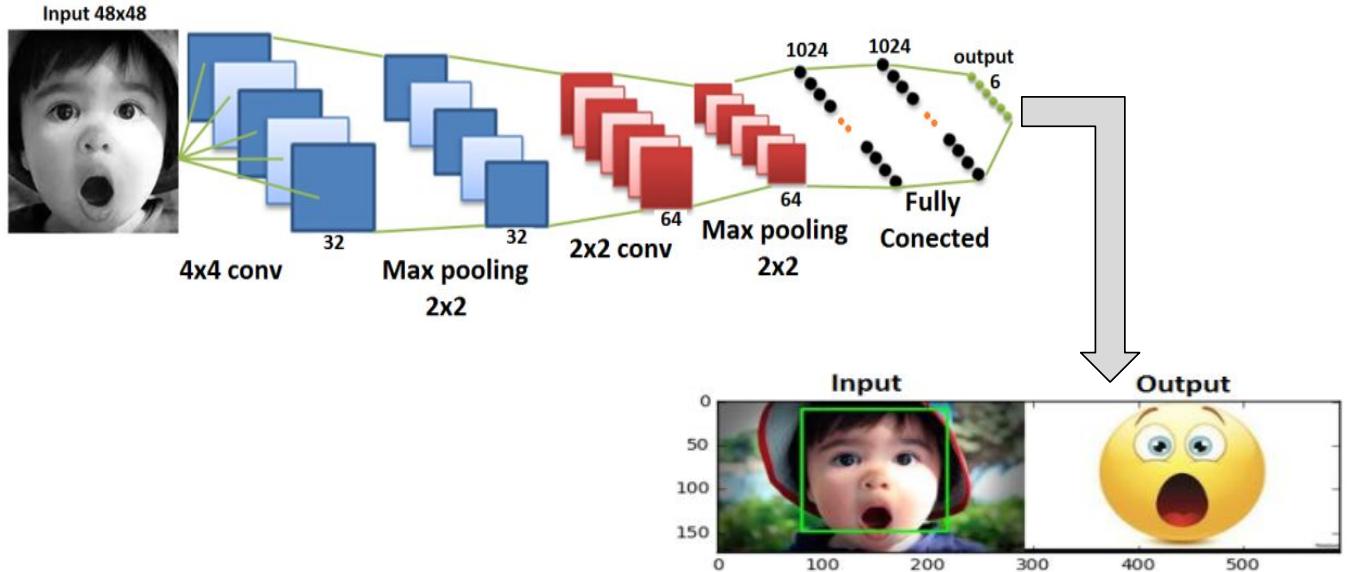


Figura 3.4: Arquitectura gráfica del modelo propuesto.

*Fuente: Propio.*

La tabla 3.7 muestra de arriba para abajo, la representación secuencial de las capas dentro de la arquitectura propuesta antes mencionada. Sin embargo, La figura 3.4 muestra de una forma mas detallada cada una de las capas que integran la arquitectura, construyendo de esta forma una red neuronal convolucional. También se muestra en esta figura, la salida final obtenida despues de realizar una consulta sobre el modelo, resaltando el rostro detectado en la imagen de entrada, y relacionando la expresión facial que este tiene con una imagen caricaturizada. El código fuente y los conjuntos de datos estan disponibilizados en link 1 y link 2 respectivamente.

### 3.4. Descripción de la Capas de la Arquitectura

Dentro de cada capa perteneciente a la red neuronal convolucional, se lleva a cabo operaciones de multiplicación de matrices, agrupación de regiones, etc, con el objetivo de extraer o resaltar características importantes de las cuales la red pueda aprender patrones y filtros que se adapten mejor al problema, de tal forma que pueda dar respuestas precisas a entradas futuras.

Siguiendo el orden de capas de la arquitectura propuestas, presentada en la sección 3.3, se describe el comportamiento de cada uno de ellos de la siguiente forma:

- **Primera capa convolución.** Cuenta con 32 filtros (mapa de características) de tamaño 4x4 píxeles. Esta capa tiene como objetivo extraer características de alto nivel. La figura 3.5 muestra el conjunto de imágenes generadas después de aplicar los 32 filtros de convolución sobre la imagen de entrada, estas imágenes muestra claramente que en esta capa, la arquitectura se encarga de aprender filtros que sean capaces de extraer bordes y partes fundamentales del rostro, tales como: los ojos, boca, nariz y cejas. Esta

primera capa de convolución genera una imagen de salida por cada filtro, siendo así, 32 nuevas imágenes las entradas para la siguiente capa.

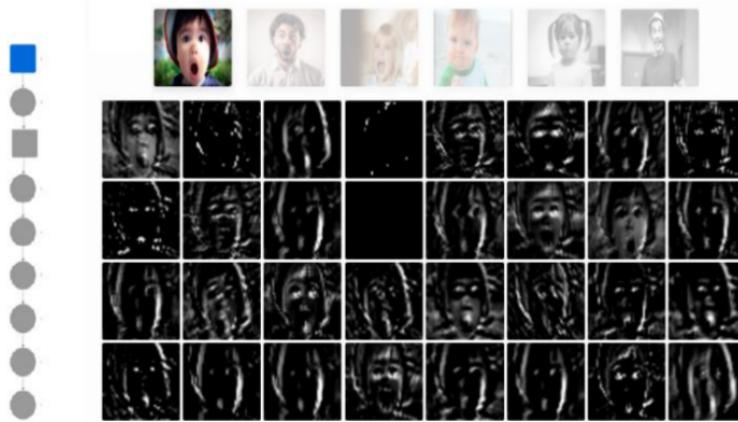


Figura 3.5: Imágenes de salida después de la primera convolución.

*Fuente: Propio.*

- **Primera capa de *pooling* o sub-muestreo.** Recibe como parámetros de entrada, las 32 imágenes generadas a partir de la primera capa de convolución. Su función es la de reducir características redundantes mediante la agrupación de píxeles y elección del mejor de entre ellos, esta agrupación se realiza en sub-regiones no solapadas de la matriz, de dimensiones  $2 \times 2$ . Despues de obtener estas sub-regiones, se obtiene el máximo pixel de entre ellas para ser la salida de dicha sub-región. La figura 3.6 muestra imágenes semejantes a las obtenidas por la primera capa de convolución, esto se debe a que esta capa no realiza ninguna alteración en el valor de los píxeles, mas al contrario, esta encargada de reducir las dimensiones de las imágenes de entrada, eliminando los píxeles menos significativos. Debido a que se aplica este tipo de operaciones a cada imagen de entrada de esta capa, su número de imágenes de salida sera igual a su número de entradas.

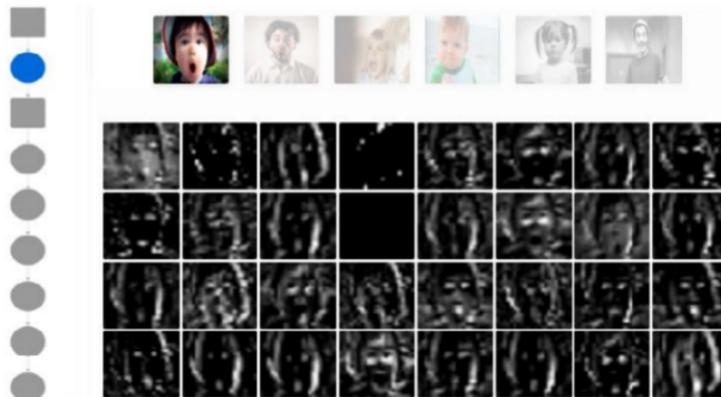


Figura 3.6: Imágenes de salida después del primer sub-muestreo.

*Fuente: Propio.*

- **Segunda capa de convolución.** Cuenta con 64 filtros y recibe como parámetros de entrada las imágenes generadas a partir de la primera capa de sub-muestreo. A diferencia de la primera capa de convolución, su función es la de extraer y detectar características de más bajo nivel. La figura 3.7 muestra las 64 imágenes generadas a partir los filtros de esta capa, en ellas se puede observar de manera más abstracta ciertos puntos y rectas que representan partes del rostro de la imagen de entrada. Como se menciono anteriormente, esta capa origina 64 imágenes de salida, cada una con un tamaño de  $21 \times 21$  píxeles.

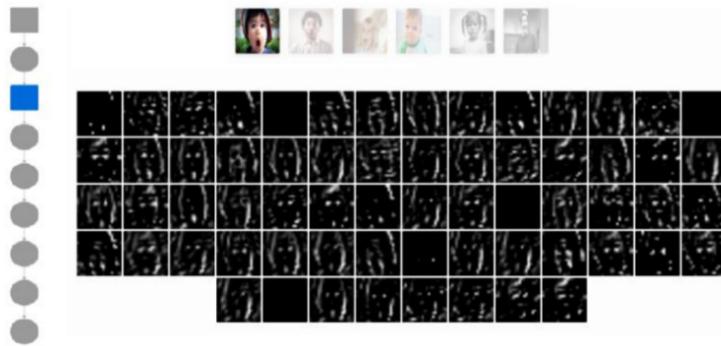


Figura 3.7: Imágenes de salida después de la segunda convolución.

*Fuente: Propio.*

- **Segunda capa de *pooling* o sub-muestreo.** Recibe como parámetros de entrada las imágenes generadas a partir de la segunda capa de convolución. Similar que la primera capa de sub-muestreo, es la encargada de eliminar características no relevantes de las imágenes de entrada, mediante la agrupación y selección del mejor pixel, las sub-regiones de agrupación son de dimensiones  $2 \times 2$ . La figura 3.8 muestra las 64 imágenes generadas a partir de esta capa, donde cada una de estas nuevas imágenes son de tamaño  $10 \times 10$  píxeles. Segun esta figura se puede observar que el rostro se distorsiona y pierde su forma original, siendo estas, a lo que se conoce como características de bajo nivel.

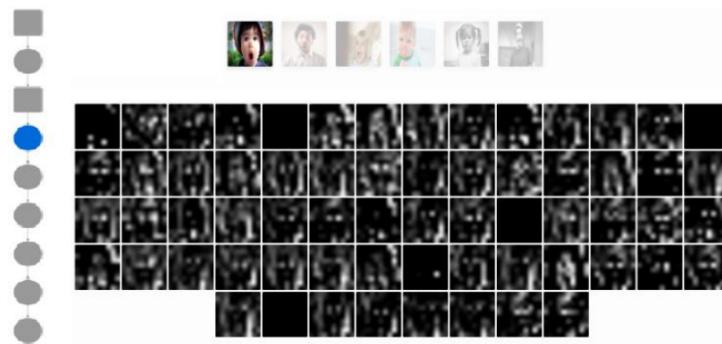


Figura 3.8: Imágenes de salida después del segundo sub-muestreo.

*Fuente: Propio.*

- **Capas totalmente conectadas.** Recibe como parámetros de entrada las imágenes generadas a partir de la segunda capa de sub-muestreo, las cuales representan las características a bajo nivel pertenecientes a la imagen de entrada original. El objetivo de esta capa, es relacionar cada una de estas características por medio de las 2 capas totalmente conectadas con 1024 neuronas cada una, donde cada neurona es una imagen que representa alguna característica encontrada en las capas anteriores. Al final de la arquitectura, se utiliza la función de normalización *softmax* para generar 6 salidas, donde cada una de ellas representa a una expresión facial. La función de normalización es la encargada de asignar una probabilidad a cada una de estas salidas, siendo nuestra salida final, aquella que presente la mayor probabilidad.

### 3.5. Parámetros de la Arquitectura

El número de parámetros que contiene una determinada arquitectura, depende mucho del número de capas, filtros, y neuronas en la capa totalmente conectada. Conocer la cantidad de parámetros en una red, es de gran utilidad, debido a que gracias a ellos podemos hacernos una idea de cuan difícil o fácil es entrenar esta. Por tanto, mientras una red o arquitectura tenga mas parámetros, esta será más costosa, obviamente, debido a la gran cantidad de parámetros por optimizar. Sin embargo, la cantidad de parámetros no necesariamente refleja la robustés de una red para una determinada tarea. Por ejemplo, puede existir problemas que con pocos parámetros a optimizar sean mejores que considerando complejas arquitecturas, o de forma viceversa.

Una manera más fácil de entender el comportamiento de la optimización de parámetros, es ver esta red como una función, donde, las salidas o etiquetas de cada entrada son el resultado de la función y cada variables son las entradas, siendo el principal objetivo, optimizar todas las constantes que multiplican a las variables, de tal manera que la función se ajuste para todas las entradas y de las salidas correctas.

La tabla 3.8 muestra la cantidad de parámetros obtenidos por la red. Un detalle muy importante, es que la capas de sub-muestreo no contienen ningún tipo de parámetro a optimizar, esto es debido a que la función de agrupación es fija y no necesita ser modificada. Sin embargo, en la capa de convolución los filtros inicialmente son ceros, y son modificados en cada etapa de entrenamiento, lo mismo ocurre con las neuronas de las últimas capas totalmente conectadas. La cantidad total de parámetros es 7,620,199, distribuidos de la siguiente forma:

- Primera capa con 32 convoluciones: 544 parámetros
- Segunda capa con 64 convoluciones: 8256 parámetros
- Primera capa totalmente conectada: 6554624 parámetros
- Segunda capa totalmente conectada: 1049600 parámetros
- función de normalización Softmax: 7175 parámetros

Layer (type)	Output Shape	Param #	Connected to
convolution2d_7 (Convolution2D)	(None, 45, 45, 32)	544	convolution2d_input_4[0][0]
maxpooling2d_7 (MaxPooling2D)	(None, 22, 22, 32)	0	convolution2d_7[0][0]
convolution2d_8 (Convolution2D)	(None, 21, 21, 64)	8256	maxpooling2d_7[0][0]
maxpooling2d_8 (MaxPooling2D)	(None, 10, 10, 64)	0	convolution2d_8[0][0]
dropout_4 (Dropout)	(None, 10, 10, 64)	0	maxpooling2d_8[0][0]
flatten_4 (Flatten)	(None, 6400)	0	dropout_4[0][0]
dense_7 (Dense)	(None, 1024)	6554624	flatten_4[0][0]
dense_8 (Dense)	(None, 1024)	1049600	dense_7[0][0]
dense_9 (Dense)	(None, 7)	7175	dense_8[0][0]

Total params: 7,620,199  
Trainable params: 7,620,199

Tabla 3.8: Número de parámetros de la arquitectura propuesta.

### 3.6. Entrenamiento de la Red

Se abordó esta tarea como un problema de aprendizaje supervisado. Como se relató en la sección 2.7, este tipo de aprendizaje requiere como datos de entrada, imágenes con una respectiva etiqueta, la cual indica la expresión facial a la que representa dicha entrada. Por lo tanto, descrito de una manera muy superficial y general, el proceso de entrenamiento consiste en ingresar imágenes de entradas con sus respectivas etiquetas a la red e intentar que esta aprenda a modelar sus pesos de tal manera que alcance las salidas deseadas. Este proceso se realiza con la ayuda del algoritmo de aprendizaje *backpropagation*, el cual propaga el error final obtenido hacia las neuronas de las capas anteriores que colaboraron en la obtención de dicho resultado.

Como se mencionó en el anterior párrafo, la etapa de entrenamiento consiste en ingresar todos los datos y esperar que la red aprenda de ellos por medio de algoritmos de aprendizaje. Sin embargo, generalmente nunca es suficiente una sola iteración de este proceso, por lo que, se repite dicho proceso  $x$  veces, siendo  $x$  más conocido como el número de *épocas*. Debido a que este tipo de algoritmos son basados en prueba y error, el entrenamiento se realiza hasta que la red comience a dar señales de sobreajuste de pesos, es decir, el aprendizaje ya no mejora, más bien empeora. En conclusión, los resultados no necesariamente serán mejores mientras más sea el número de épocas, como se mencionó antes, este tipo de abordaje(*deep learning*) en la actualidad no tiene un estudio previo que indique la cantidad de parámetros, capas, épocas y funciones que una red deba tener, por lo tanto, la respuesta está en la cantidad de experimentos que se realicen, pero debido al poder computacional y tiempo que este requiere, es una limitante para todos los investigadores en el mundo.

Para los experimentos realizados en la fase de entrenamiento, orientados a resolver el problema planteado (reconocimiento de expresiones faciales), se siguieron los pasos antes descritos y se observó que después de la época 100 la red deja de aprender. Esta descripción se muestra de manera más detallada (por cada base de datos experimentada) en la sección 3.10.

### 3.7. Pruebas al Modelo Creado

La fase de entrenamiento da origen a un archivo con extensión .h5, el cual almacena los pesos de la red. Ya con este archivo en mano, se realiza la siguiente etapa, que consiste en las pruebas al modelo creado. Estas pruebas o consultas siguen la siguiente secuencia de pasos:

- Primero, se lee una imagen como dato de entrada, de dimensiones mayor o igual a 48x48 píxeles. La imagen puede ser a colores o no, el algoritmo internamente lo procesa como una imagen en escala de gris.
- Segundo, para reducir la cantidad de información irrelevante en la imagen (partes de la imagen que no pertenecen a un rostro: cuerpo, fondo, etc), se utiliza el detector de rostros *haar cascade*, obteniendo así, cuatro coordenadas que delimitan el rostro en la imagen de entrada.
- Tercero, ya con las cuatro coordenadas obtenidas por el algoritmo de detección, se recorta la imagen. Si la imagen recortada resulta ser de tamaño menor que  $48 \times 48$  píxeles, esta se redimensiona a dicho tamaño. Caso contrario la imagen se mantiene tal cual.
- Cuarto, la imagen obtenida en el paso anterior es pasada como dato de consulta al modelo. Este devuelve un valor entero en el intervalo  $[0 - 6]$ , donde cada uno de esos numeros representa una expresion facial. Finalmente, se muestra como imagen de salida la imagen de consulta con una figura caricaturizada que esta relacionada con el número obtenido.

La figura 3.9 muestra los pasos 3 y 4 mencionadas en los párrafos anteriores.



Figura 3.9: Secuencia de pasos en la fase de pruebas.

Fuente: Propio.

### 3.8. Recopilación de Imágenes de Expresiones Faciales.

En la actualidad el internet nos ha abastecido con grandes cantidades de información. Dentro de estas, podemos encontrar comunidades dedicadas a la investigación que proveen bases de datos de imágenes etiquetadas relacionadas con problemas específicos, tales como: el reconocimiento de placas de automóviles, clasificación de imágenes o como en nuestro caso,

reconocimiento de expresiones faciales. Estas bases de datos son utilizadas por toda la comunidad científica, publicando y comparando sus resultados con trabajos previos y acercandoce más y más a un margen de error mínimo. Generalmente estas bases de datos cuentan con imágenes cuidadosamente seleccionadas, intentando cubrir todos los casos posibles que se podrían presentar en escenas de la vida real.

Como se mencionó en el párrafo anterior, en este trabajo, la recopilación de las imágenes de expresiones faciales se obtuvo de 2 fuentes secundarias de información (internet) en los cuales los datos están pre-elaborados. La primera base de datos utilizada es contiene imágenes de  $48 \times 48$  píxeles de tamaño, mientras que la segunda de  $640 \times 480$  o  $640 \times 490$  píxeles. Para la fase de prueba también se utilizaron imágenes aleatorias de internet, dentro de estas imágenes se consideran personas del mundo real y caricaturizadas (ver figura 3.10).

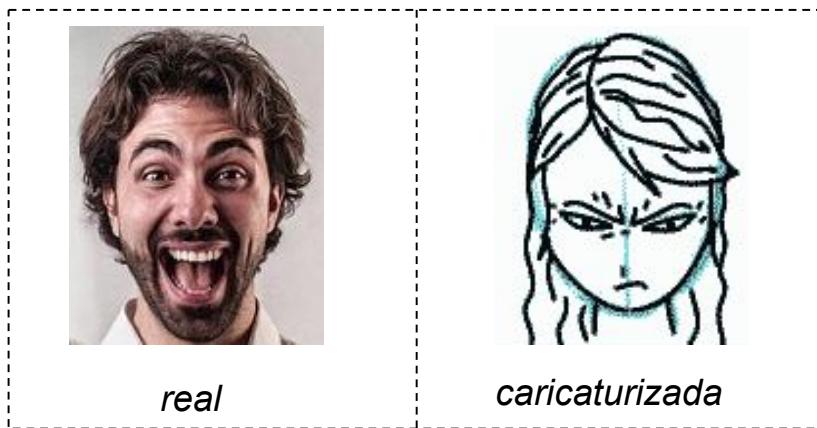


Figura 3.10: Ejemplo de un rostro real y caricaturizado.

*Fuente: Internet.*

## 3.9. Base de Datos

Para la realización de experimentos se utilizaron 3 bases de datos, dos de ellas se encuentran disponibles en internet (*FER103* y *CK+*) y una tercera que es resultado de la unión de estas dos anteriores. Se optó por considerar esta tercera base de datos, debido a que el abordage de *deep learning* trabaja mejor mientras tenga mas datos para la fase de entrenamiento.

### 3.9.1. FER2013

Es una base de datos del sitio web *Kaggle*, destinada para el concurso de reconocimiento de expresiones faciales. Fue preparada por Pierre-Luc Carrier y Aaron Courville, como parte de un proyecto de investigación en curso. Esta clasificada en 7 categorías: molesto, disgustado, miedo, feliz, triste, sorpresa y neutro, sin embargo, en este trabajo se optó por unir la categoría enojado y disgustado debido a las similitudes que ambas tienen.

Este conjunto de datos está disponible como archivos *train.csv* y *test.csv*, los cuales contienen 2 columnas: *emotion* y *pixels*. La columna *emotion* contiene un código numérico en

el rango de 0 a 6 inclusive, para la expresión que esta presente en la imagen. La columna *pixels* contiene un string de números que representa los valores para cada pixel dentro de la imagen. El conjunto de entrenamiento consiste de 28,709 ejemplos, mientras que los datos de prueba son 3589. La figura 3.11 muestra ejemplos de imágenes de este conjunto de datos.



Figura 3.11: Imágenes de la base de datos FER2013

*Fuente: Kaggle.*

### 3.9.2. CK+

La base de datos CK+ (Cohn-Kanade) posee imágenes de expresiones faciales frontales de 210 diferentes personas. Estas imágenes tienen una resolución original de 640x490 y 640x480 píxeles. Debido a que solo se encontraron imágenes pertenecientes a esta base de datos y sin ninguna etiqueta que relacione la expresión facial que representa cada una de ellas, en este trabajo se optó por realizar la etiquetación de forma manual (basandonos en el trabajo de Paul Ekman), obteniendo así 3289 imágenes, transformandolas a imágenes en escala de gris con dimensiones de 48x48 píxeles. Similar que la anterior base de datos, se consideraron 6 categorías (enojado, miedo, feliz, triste, sorprendido y neutro). Se utilizó el 80 % del total de imagenes para la fase de entrenamiento (2966 imágenes) y el 20 % para la fase de pruebas (323 imágenes). La figura 3.12 muestra ejemplos de imágenes de este conjunto de datos.

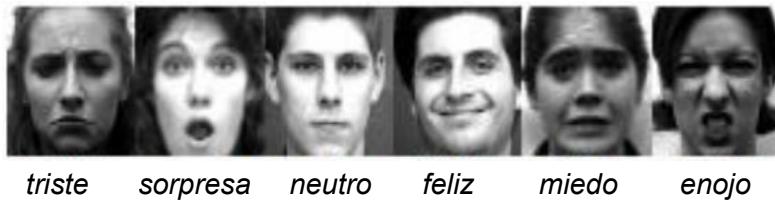


Figura 3.12: Imágenes de la base de datos CK+

*Fuente: Base de datos CK+.*

### 3.9.3. FER2013 - CK+

Esta base de datos resulta de la unión de las dos anteriores (Fer2013 y CK+), obteniendo así un total de 39176 imágenes en escala de gris de dimensiones de 48x48 píxeles. El conjunto de entrenamiento es el 80 % del total (35264 imágenes), mientra que el conjunto de prueba es el 20 % (3912 imágenes). Se realizo esta unión de datos con la finalidad de obtener un modelo

más robusto, ya que las técnicas de *deep learning* dependen mucho de la cantidad de datos para el entrenamiento de un modelo.

## 3.10. Resultados Experimentales

Los experimentos se realizaron sobre los tres conjuntos de datos mencionados en la sección anterior. En esta sección se muestran los resultados de dichos experimentos uno por uno, donde se podrán apreciar los niveles de precisión alcanzados por cada categoría - expresión facial, así como sus matrices de confusión, las cuales ayudan para un mejor entendimiento e interpretación de los resultados. También son mostrados por medio de gráfico, el aprendizaje de la red medida en términos de precisión durante cada época.

Para una mejor comprensión e interpretación de los resultados, se definen las siguientes palabras: *precisión* es la fracción de instancias (datos de entrada) que alcanzaron un resultado correcto entre el número de instancias totales, *recall* es la cantidad de instancias correctas recuperadas sobre la cantidad total de instancias relevantes (falsos positivos y falsos negativos). La figura 3.13 muestra de una manera más sencilla ambas definiciones. Por otro lado *f1-score* es la media armónica de la precisión y el *recall*, la ecuación 3.1 muestra su representación matemática. *Falsos positivos* o *false positives* en inglés, son el conjunto de instancias cuyas respuestas se creen que son correctas cuando en realidad están erradas. *Falsos negativos* o *false negatives* a diferencia del anterior, son el conjunto de instancias que creen que están errados cuando en realidad son correctos. *verdaderos positivos* y *verdaderos negativos* son simplemente instancias que son o bien correctas o erróneas.

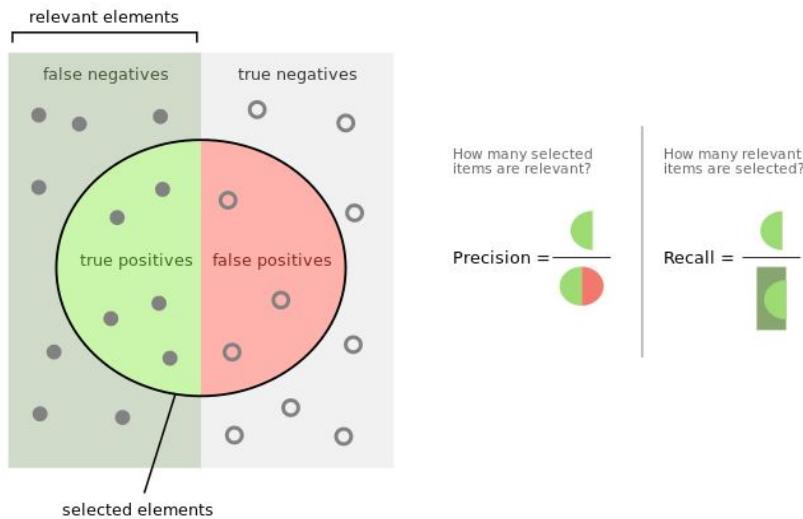


Figura 3.13: Precision y recall.

Fuente: *Precision and recall*, Wikipedia.

$$F1 = 2 \times \frac{precision \times recall}{precision + recall} \quad (3.1)$$

### 3.10.1. FER2013

	precision	recall	f1-score	support
Enojado	0.46	0.58	0.51	546
Miedo	0.52	0.36	0.43	528
Feliz	0.74	0.75	0.75	879
Triste	0.43	0.40	0.41	594
Sorprendido	0.71	0.77	0.74	416
Neutro	0.52	0.54	0.53	626
avg / total	0.57	0.57	0.57	3589

Tabla 3.9: Resultados obtenidos - FER2013

En la tabla 3.9 se muestra los resultados obtenidos durante los experimentos realizados en la base de datos FER2013. Dentro de los niveles de precisión y *recall*, se observa que las categorías 'feliz' y 'sorpresa' obtienen mejores resultados que las demás, alcanzando un nivel de precisión de 74 % y 71 % respectivamente. Por otro lado, la categoría 'enojado' alcanza 46 %, siendo la más baja de entre todas. El resultado general alcanzado en este conjunto de datos es de 57 %.

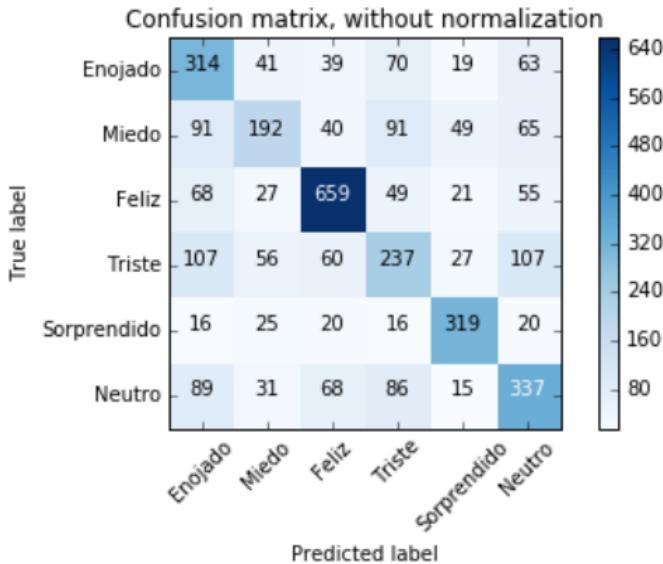


Figura 3.14: Matriz de confusión, precisión del Test - FER2013

Gracias a la matriz de confusión (figura 3.14) se puede descubrir con exactitud la cantidad de datos que fallan y aciertan por categoría. Como se mencionó anteriormente, la categoría 'feliz' es la que presenta mayor precisión, acertando 659 imágenes de las 879 en total, también podemos observar que la red confunde esta categoría en mayor cantidad con la categoría 'neutro', siendo en total 68 imágenes de expresiones faciales 'neutrales' confundidas con expresiones de la categoría 'feliz'. Otro detalle muy importante que nos muestra esta imagen, es que la red confunde en grandes cantidades las categorías 'triste' con 'enojado' y 'neutro', lo que nos

da a entender que deberíamos poner mas enfasis e importancia en imágenes pertenecientes a estas categorías.

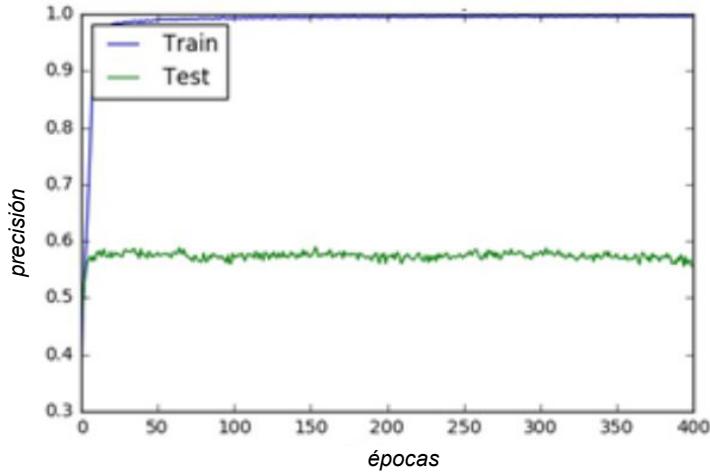


Figura 3.15: Precisión durante el proceso de entrenamiento y prueba (%) – FER2013

La figura 3.15 muestra la precisión alcanzada por la red durante el proceso de entrenamiento y pruebas en el transcurso de cada época. La linea azul muestra el crecimiento del aprendizaje con los datos de entrenamiento, se puede observar que entre la época 50 y 100 llega a alcanzar el 100 % de precisión, sin embargo esto sucede por que se realiza este cálculo con datos que ya fueron visualizados por la red. La linea verde muestra la precisión alcanzada con los datos de prueba, alcanzando un 57 %, esto sucede por que los datos de prueba son datos nuevos para la red, datos que nunca fueron vistos durante su etapa de entrenamiento.

### 3.10.2. CK+

	precision	recall	f1-score	support
Enojado	0.91	0.76	0.83	66
Miedo	0.76	1.00	0.86	25
Feliz	1.00	0.99	0.99	83
Triste	0.87	1.00	0.93	48
Sorprendido	1.00	0.90	0.95	58
Neutro	0.78	0.84	0.81	43
avg / total	0.91	0.91	0.91	323

Tabla 3.10: Resultados obtenidos - CK+

La tabla 3.10 muestra los resultados obtenidos durante los experimentos realizados en la base de datos CK+. Dentro de los niveles de precisión y *recall*, se observa que las categorías 'feliz' y 'sorpresa' obtienen mejores resultados que las demás (de igual forma que los resultados en la anterior base de datos), alcanzando un nivel de precisión de 100 % y 100 %

respectivamente. Por otro lado, la categoría 'miedo' alcanza 46 %, siendo la mas baja de entre todas. El resultado general alcanzado en este conjunto de datos es de 91 %.

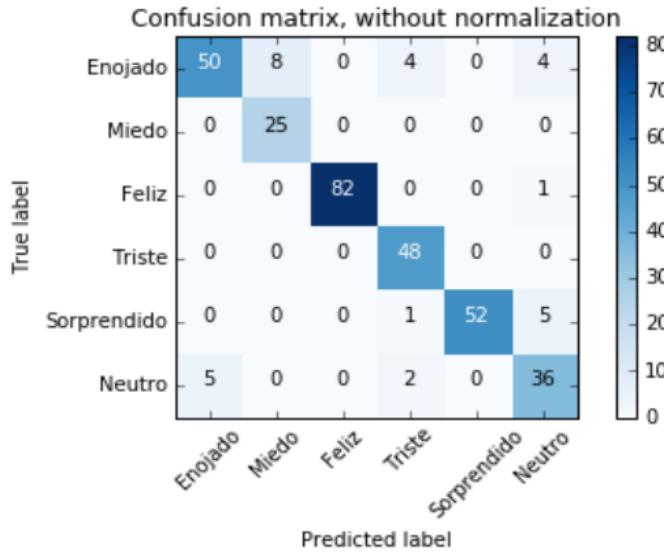


Figura 3.16: Matriz de confusión, precisión del Test - CK+

La figura 3.16 muestra la matriz de confusión correspondiente a los resultados obtenidos para esta base de datos. Como se mostró en la tabla anterior, se puede observar que de las 82 muestras de la categoría 'feliz', todas son acertadas, alcanzando así el 100 % de precisión mencionado anteriormente. También se puede observar que en la categoría 'miedo' de las 88 imágenes en total, confunde 8 de ellas con la categoría 'enojado'. Otra de las categorías que también es confundida con las demás, es 'neutro', donde de 46 imágenes en total, 5 son confundidas con 'sorpresa', 4 con 'enojo' y 1 con 'feliz'.

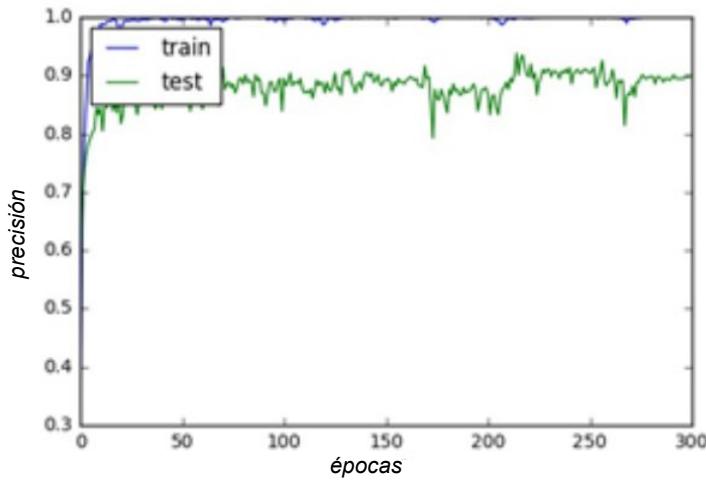


Figura 3.17: Precisión durante el proceso de entrenamiento y prueba (%) - CK+

La figura 3.17 muestra la precisión alcanzada por la red durante el proceso de entrenamiento y pruebas en el transcurso de cada época. La linea azul representa la precisión

alcanzada con los datos de entrenamiento, donde se observa claramente que este alcanza el 100 % de precisión aproximadamente en la época 50, sin embargo, también esta curva presenta oscilaciones en ciertas épocas, bajando ligeramente su nivel de precisión. La linea verde representa la precisión alcanzada con el conjunto de pruebas, el cual muestra el pico más alto de la curva entre la época 200 y 300, siendo este número de épocas el apropiado para obtener la mejor precisión posible.

### 3.10.3. FER2013 - CK+

	precision	recall	f1-score	support
Enojado	0.57	0.50	0.53	612
Miedo	0.48	0.46	0.47	553
Feliz	0.75	0.79	0.77	962
Triste	0.44	0.48	0.46	642
Sorprendido	0.77	0.72	0.74	474
Neutro	0.53	0.55	0.54	669
avg / total	0.60	0.60	0.60	3912

Tabla 3.11: Resultados obtenidos - FER2013 - CK+

La tabla 3.11 muestra los resultados obtenidos durante los experimentos realizados en la unión de las bases de datos FER2013 y CK+. Dentro de los niveles de precisión y *recall*, se observa que las categorías 'feliz' y 'sorpresa' obtienen mejores resultados que las demás (de igual forma que en las anteriores bases de datos), alcanzando un nivel de precisión de 75 % y 77 % respectivamente. Por otro lado, la categoría 'triste' alcanza 44 %, siendo la mas baja de entre todas. El resultado general alcanzado en este conjunto de datos es de 60 %.

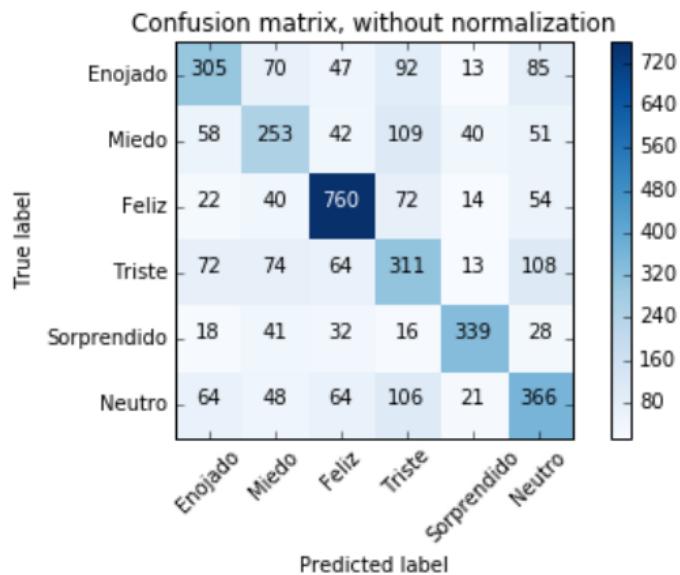


Figura 3.18: Matriz de confusión, precisión del Test FER2013 - CK+

La figura 3.18 muestra la matriz de confusión correspondiente a la unión de las dos bases de datos antes mencionadas. En ella se puede observar que muchas categorías son confundidas por las otras en cantidades considerables, sin embargo la categoría 'triste' es la que más errores comete, siendo 109 y 106 imágenes confundidas con las categorías 'miedo' y 'neutro' respectivamente.

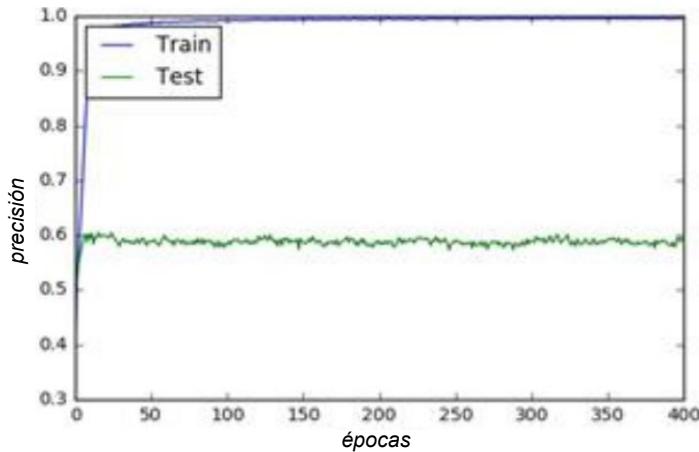


Figura 3.19: Precisión durante el proceso de entrenamiento y prueba (%) FER2013 - CK+

En la figura 3.19, similar que en las dos anteriores gráficas de precisión correspondientes a las anteriores bases de datos, se puede observar que con los datos de entrenamiento se obtiene un nivel de precisión del 100%, sin embargo, con los datos de prueba se genera una curva con pequeñas oscilaciones durante el transcurso de las épocas, por lo tanto se concluye que utilizando un número de épocas mayor o igual a 20 se obtendrá el máximo nivel de precisión.

# Resultados Generales

Se obtuvo un nivel de precisión de 91 % en los experimentos realizados con la base de datos CK+ (sub-sección 3.9.2), 57 % con FER2013 (sub-sección 3.9.1) y 60 % con la unión de ambas bases de datos (sub-sección 3.9.3), estando 14 % debajo del nivel de precisión del primer lugar del concurso mundial de reconocimiento de expresiones faciales organizado por *Kaggle* (utilizando la base de datos FER2013). Se presume que dichos resultados no alcanzaron ni superaron métodos del estado del arte, debido a la cantidad de experimentos realizados por las limitaciones de *hardware* que se presentaron en el desarrollo de este trabajo (uso de CPU más no de GPU en la fase de entrenamiento). Sin embargo, los resultados obtenidos con la base de datos CK+ son prometedores, debido a la variedad de imágenes de expresiones faciales que este conjunto de datos posee.

# CONCLUSIONES

- En este trabajo, desarrollamos un clasificador para el reconocimiento de expresiones faciales considerando seis categorías (alegre, neutro, feliz, triste, enojado, sorpresa).
- Las bases de datos usadas en este trabajo Fer2013 3.9.1 y CK+ 3.9.2, fueron estandarizados en tamaño y posteriormente normalizados en valor para reducir la varianza en los datos.
- Adicionalmente, se creó una tercera base de datos llamado Fer2013-CK+ 3.9.3, como resultado de la unión de las bases de datos Fer2013 y CK+.
- Desarrollamos una arquitectura de red neuronal convolucional para el reconocimiento de expresiones faciales 3.3.
- La arquitectura usada para crear el modelo para el reconocimiento de expresiones faciales fue el resultado de tres arquitecturas propuestas, siendo seleccionada la arquitectura con mejores resultados en los experimentos 3.10.
- Cada una de las tres arquitectura propuestas, tuvo una configuración diferente de los elementos principales que componen una red neuronal convolucional (número de capas de convolución, tamaño de los filtros de convolución, número de capas de pooling, operación de pooling, número de capas de neuronas totalmente conectadas, número de neuronas en la capa totalmente conectada) basada en trabajos de la literatura de *deep learning* mostrados en 3.10..
- Se realizó satisfactoriamente los experimentos en las tres bases de datos utilizadas Fer2013, CK+ y Fer2013-CK+ 3.10.
- Se evaluó los resultados obtenidos por el modelo creado en las tres bases de datos Fer2013, CK+, Fer2013-CK+. Obteniendo un resultado de 90 % de precisión en la base de datos CK+, mientras que en las bases de datos Fer2013 y Fer2013-CK+ se obtuvo 57 % y 60 % de precisión respectivamente (más detalles ver Tablas 3.10, 3.9, 3.11 ).
- Adicionalmente, se tenía la idea de alterar las imágenes de los conjuntos de datos con rotaciones variadas, iluminación, etc (imágenes sintéticas). Sin embargo, esto no representaría las condiciones de las imágenes del mundo real, por lo que nosotros optamos por recolectar un pequeño conjunto de imágenes de internet con contenido variado (fondo con diferente iluminación, oclusión parcial del rostro, posición rotada en algunos rostros,

imagen de animes) que si representan en su mayoria las condiciones de las imágenes del mundo real. Así, se hicieron pruebas en este conjunto de imágenes de internet. Los resultados obtenidos (pruebas satisfactorias - pruebas fallidas) son mostrados en B.1.

# RECOMENDACIONES

Por la experiencia adquirida en la realización de este proyecto de investigación, se lista un conjunto de sugerencias útiles para los lectores.

- Se recomienda obtener un equipamiento de *hardware* adecuado para trabajar con *deep learning*. Principalmente la adquisición de *GPUs* que facilitarán el entrenamiento de redes neuronales profundas por su poder de paralelización en operaciones con matrices. Caso no exista la posibilidad de adquirir un equipamiento propio para *deep learning*, es recomendado usar una computadora con capacidad minima de 8GB de RAM. Sin embargo, su capacidad estará limitada a redes neuronales con menor profundidad. Existe una opción adicional, alquilar servidores en la nube especializados en el entrenamiento de arquitecturas de *deep learning*. La desventaja de estos servicios es que son de pago.
- En el proceso de investigación, se recomienda que la información extraída sea de fuentes confiables. Para ello, sugerimos que accedan a material de investigación (papers, artículos científicos y otros) de instituciones prestigiosas como la IEEE, ACM, SPRINGER y otros.
- Respecto a las herramientas para implementación, recomendamos usar Python como lenguaje de programación por las facilidades que brinda y por su uso concurrido a nivel mundial. Además, de tener muchas bibliotecas que facilitan el trabajo con *deep learning*.

# TRABAJOS FUTUROS

En este trabajo se abordo el reconocimiento de expresiones en imágenes estáticas por medio de redes neuronales convolucionales. Por lo que se tiene pendiente como trabajos futuros la realización de un clasificador de expresiones faciales en tiempo real. Para lo cual tenemos pensado en la utilización de técnicas mas avanzadas para la fase de detección de rostros. Redefinir la arquitectura propuesta para una arquitectura mas profunda. así, como la utilización técnicas de *data augmentation* para incrementar la base de datos y aumentar la variabilidad de las imágenes y *transfer learning* para facilitar la fase de entrenamiento y mejorar el modelo.

# BIBLIOGRAFÍA

- [1] T. R. Almaev and M. F. Valstar. Local gabor binary patterns from three orthogonal planes for automatic facial expression recognition. In *Affective Computing and Intelligent Interaction (ACII), 2013 Humaine Association Conference on*, pages 356–361. IEEE, 2013.
- [2] M. Alonso, A. Díaz, M. Alonso, and A. Aguado. *Personas con discapacidad: perspectivas psicopedagógicas y rehabilitadoras*. Manuales (Siglo XXI de España Editores).: Psicología. Siglo XXI de España, 2005.
- [3] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.
- [4] D. Ciregan, U. Meier, and J. Schmidhuber. Multi-column deep neural networks for image classification. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3642–3649. IEEE, 2012.
- [5] C. Clark and A. Storkey. Training deep convolutional neural networks to play go. In *International Conference on Machine Learning*, pages 1766–1774, 2015.
- [6] CS231n. Convolutional Neural Network for Visual Recognition. <http://cs231n.github.io/convolutional-networks/#norm>, 2016. Last access: 2016-07-13.
- [7] deeplearning4j. Early Stopping. <https://deeplearning4j.org/earlystopping#early-stopping>, 2016. Last access: 2016-10-30.
- [8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.
- [9] P. Ekman. Facial expression and emotion. *American psychologist*, 48(4):384, 1993.
- [10] P. Ekman. Why don't we catch liars? *Social research*, pages 801–817, 1996.
- [11] P. Ekman. What scientists who study emotion agree about. *Perspectives on Psychological Science*, 11(1):31–34, 2016.
- [12] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer vision and Image understanding*, 106(1):59–70, 2007.

- [13] M. Ghayoumi and A. K. Bansal. An integrated approach for efficient analysis of facial expressions. In *Signal Processing and Multimedia Applications (SIGMAP), 2014 International Conference on*, pages 211–219. IEEE, 2014.
- [14] I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*. MIT press, 2016.
- [15] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. 2007.
- [16] S. Houben, J. Stallkamp, J. Salmen, M. Schlipsing, and C. Igel. Detection of traffic signs in real-world images: The german traffic sign detection benchmark. In *Neural Networks (IJCNN), The 2013 International Joint Conference on*, pages 1–8. IEEE, 2013.
- [17] Intelygenz. Machine Learning and application. <http://www.intelygenz.es/que-es-machine-learning-y-que-aplicaciones-tiene-dia-a-dia/>, 2016. Last access: 2016-10-27.
- [18] José Rosales Fernández. Redes Neuronales. <http://www.usmp.edu.pe/publicaciones/boletin/fia/info32/pag4.htm>, 2016. Last access: 2016-09-11.
- [19] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.
- [20] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- [21] B. W. M. A. Lehr. Backpropagation and its applications. In *Neural Network Computing for the Electric Power Industry: Proceedings of the 1992 INNS Summer Workshop*, page 21. Psychology Press, 1993.
- [22] T. M. Mitchell. Machine learning. 1997. *Burr Ridge, IL: McGraw Hill*, 45(37):870–877, 1997.
- [23] R. Padilla, C. Costa Filho, and M. Costa. Evaluation of haar cascade classifiers designed for face detection. *World Academy of Science, Engineering and Technology*, 64, 2012.
- [24] L. Prechelt. Early stopping-but when? *Neural Networks: Tricks of the trade*, pages 553–553, 1998.
- [25] P. D. Pusiol. Redes convolucionales en comprensión de escenas. B.S. thesis, 2014.
- [26] G. J. P. Restrepo Arteaga et al. Aplicación del aprendizaje profundo (deep learning) al procesamiento de señales digitales. B.S. thesis, Universidad Autónoma de Occidente, 2015.
- [27] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.
- [28] C. Sabino. Como hacer una tesis y elaborar todo tipo de escritos. *Editorial Panapo. Venezuela*, 1994.

- [29] P. Sermanet and Y. LeCun. Traffic sign recognition with multi-scale convolutional networks. In *Neural Networks (IJCNN), The 2011 International Joint Conference on*, pages 2809–2813. IEEE, 2011.
- [30] E. L. d. Silva and E. M. Menezes. Metodologia da pesquisa e elaboração de dissertação. 2001.
- [31] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural networks*, 32:323–332, 2012.
- [32] L. E. Sucar and G. Gómez. Visión computacional. *Instituto Nacional de Astrofísica, Óptica y Electrónica. México*, 2011.
- [33] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–I. IEEE, 2001.
- [34] Wikipedia. Autoencoder. <https://en.wikipedia.org/wiki/Autoencoder>, 2016. Last access: 2016-10-22.
- [35] Wikipedia. Confusion Matrix. [https://en.wikipedia.org/wiki/Confusion\\_matrix](https://en.wikipedia.org/wiki/Confusion_matrix), 2016. Last access: 2016-10-26.
- [36] Wikipedia. Función de activación Relu. [https://en.wikipedia.org/wiki/Rectifier\\_\(neural\\_networks\)](https://en.wikipedia.org/wiki/Rectifier_(neural_networks)), 2016. Last access: 2016-10-20.
- [37] Wikipedia. Función de activación Sigmoide. [https://es.wikipedia.org/wiki/Funci%C3%B3n\\_sigmoide](https://es.wikipedia.org/wiki/Funci%C3%B3n_sigmoide), 2016. Last access: 2016-10-20.
- [38] Wikipedia. Función de activación tangencial. [https://es.wikipedia.org/wiki/Tangente\\_hiperb%C3%ADlica](https://es.wikipedia.org/wiki/Tangente_hiperb%C3%ADlica), 2016. Last access: 2016-10-20.
- [39] Wikipedia. Visión por computador. [https://es.wikipedia.org/wiki/Visi%C3%B3n\\_artificial#cite\\_note-1](https://es.wikipedia.org/wiki/Visi%C3%B3n_artificial#cite_note-1), 2016. Last access: 2016-08-01.
- [40] Y. LeCun, C. Cortes, y C. Burges. Mnist: A dataset of handwritten digits. <http://yann.lecun.com/exdb/mnist/>, 2014. Last access: 2016-10-01.
- [41] L. Zhong, Q. Liu, P. Yang, B. Liu, J. Huang, and D. N. Metaxas. Learning active facial patches for expression analysis. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2562–2569. IEEE, 2012.

# **ANEXOS**

# Apéndice A

## Otros conceptos

### A.1. Matriz de Confusión

La matriz de confusión es una técnica que facilita el análisis del rendimiento de un algoritmo de clasificación. Frecuentemente, el rendimiento de un clasificador es solo analizado en términos de la *precision* (verdaderos positivos, verdaderos negativos y falsos positivos). Sin embargo, este no refleja completamente el verdadero rendimiento del clasificador. Así, comúnmente es considerado el *recall* para solucionar parcialmente este problema. El *recall* toma en cuenta los falsos negativos (repuestas predichas como falsas cuando en realidad son verdaderas). Adicionalmente, para evaluar el rendimiento final de un clasificador en forma completa se usa el *f1-score*. El f1-score toma en consideración tanto la *precision* y el *recall*, obteniendo una penalización del rendimiento del clasificador cada vez que este se equivoque [35]. Así, la matriz de confusión cumple un papel importante en el cálculo de *precision*, *recall* y *f1-score*. En la Tabla A.1 se puede observar la estructura general de una matriz de confusión. así mismo en las ecuaciones A.1 y A.2 se muestra el uso de los elementos la matriz de confusión para calcular las métricas de *precision* y *recall* respectivamente. Finalmente, la ecuación A.3 muestra el cálculo de *f1-score*.

		Valores referenciales	
		Positivos	Negativos
Valores predichos	Positivos	True Positive (TP)	False Negative (FN)
	Negativos	False Positive (FP)	True Negative (TN)

Cuadro A.1: Estructura general de la matriz de confusión

$$precision = \frac{TP}{TP + FP} \quad (\text{A.1})$$

$$recall = \frac{TP}{TP + FN} \quad (\text{A.2})$$

$$f1-score = \frac{2 \cdot precision \cdot recall}{precision + recall} \quad (\text{A.3})$$

## A.2. Machine Learning

El *machine learning* es un área de la inteligencia artificial que alberga algoritmos de aprendizaje. La definición: “Se dice que un programa de computadora aprende de la experiencia E con respecto a alguna clase de tareas T y la medida de rendimiento P, si su desempeño de tareas en T, medido por P, mejora con la experiencia E.” realizada en [22] nos explica que significa aprender para una máquina.

## A.3. Aplicaciones de Machine Learning

Los algoritmos de *machine learning* pueden ser usados para una infinidad de aplicaciones que solucionen problemas del mundo real. La amplia gama de aplicaciones en los que pueden ser utilizados los algoritmos de *machine learning* van desde motores de búsqueda, diagnósticos médicos, reconocimiento del habla y del lenguaje, robótica, etc. Se mencionan algunas de las aplicaciones con mayor tendencia que tienen como elemento principal el uso de algoritmos de *machine learning* [17]:

- Detección de rostro.
- Reconocimiento facial, de voz o de objetos.
- Motores de Busqueda.
- Anti-spam.
- Anti-virus.
- Genética.
- Predicción y pronósticos del clima.
- Comprensión de textos.
- Vehículos autónomos y robots.
- Análisis de imágenes de alta calidad.
- Análisis de datos económicos.
- Análisis de comportamiento de consumo y productividad.

## A.4. Early Stopping

Cuando se entrena una red neuronal se tiene que definir a priori cuantas iteraciones se harán sobre el conjunto de datos de entrenamiento. Depende fuertemente del número de épocas escogido si se obtendrá un buen ajuste a los datos o por el contrario podra ocurrir sobreajuste en los datos de entrenamiento.

El método de *early stopping* ayuda a resolver el problema de escoger el número de épocas en el que logra un buen ajuste a los datos de entrenamiento [7], [24].

Los pasos a seguir para la aplicación del método de early stopping son los siguientes:

- Dividir la base de datos en 2 partes: base de datos de entrenamiento y base de datos de validación.
- Para cada N épocas debemos hacer lo siguiente:
  - Evaluar el error del modelo en el conjunto de validación.
  - Parar el entrenamiento si el error de validación es mayor respecto al anterior error de validación.
  - Finalmente, se selecciona el modelo creado antes del aumento del error de validación.

En la Figura A.1 se puede observar la relación época y rendimiento en el conjunto de entrenamiento y el conjunto de validación. Así, como el punto óptimo en el que se alcanza el mejor rendimiento en el conjunto de validación.

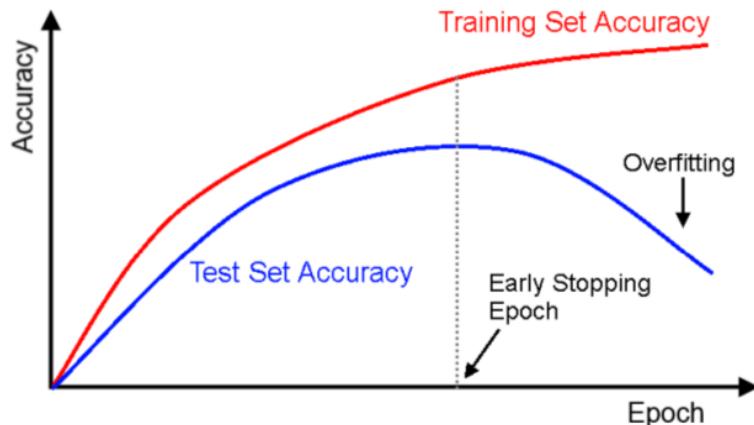


Figura A.1: Representación gráfica de Early Stopping  
Fuente: *deeplearning4j* [7]

# Apéndice B

## Testing

### B.1. Pruebas satisfactorias



Figura B.1: Test 1

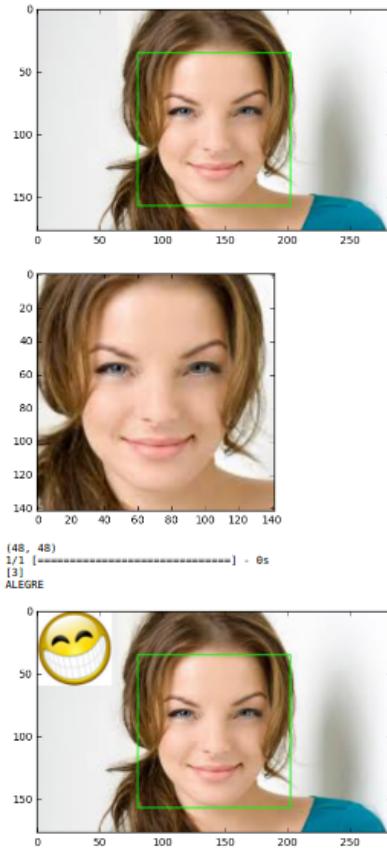


Figura B.2: Test 2

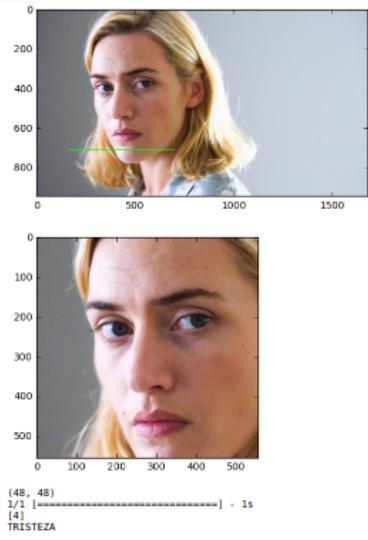


Figura B.3: Test 3

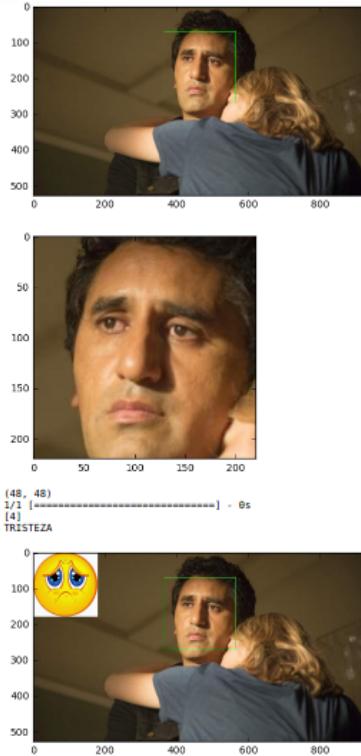


Figura B.5: Test 5

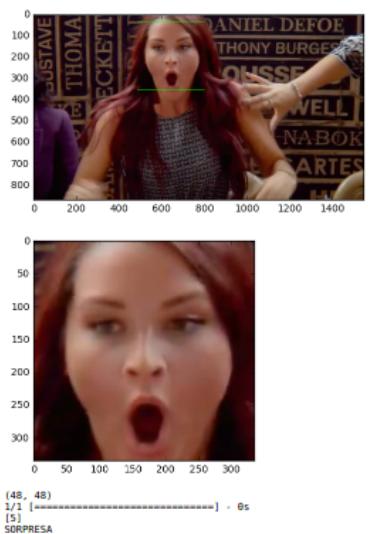


Figura B.4: Test 4

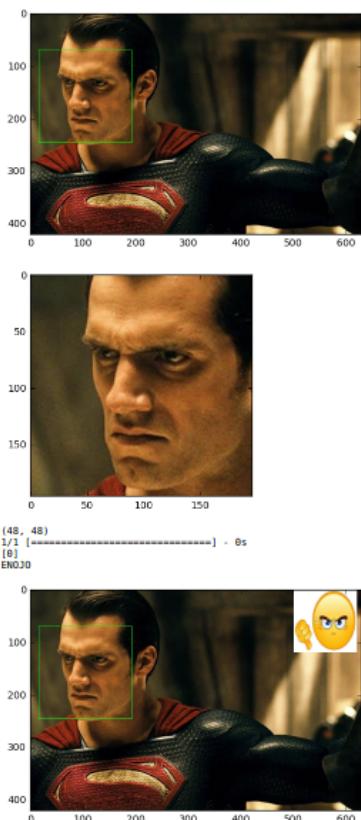
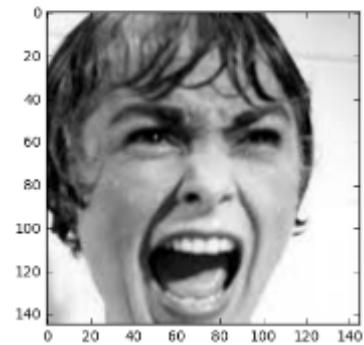
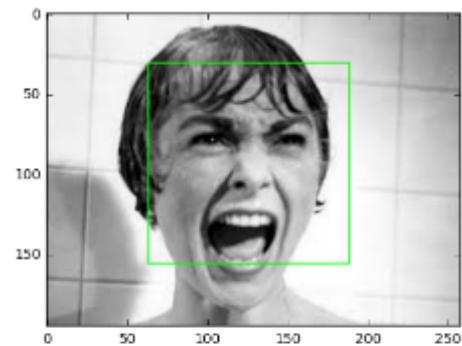


Figura B.6: Test 6



(48, 48)  
1/1 [=====] - 1s  
[2]  
MIEDO

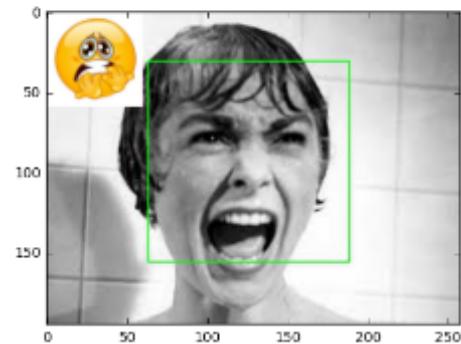
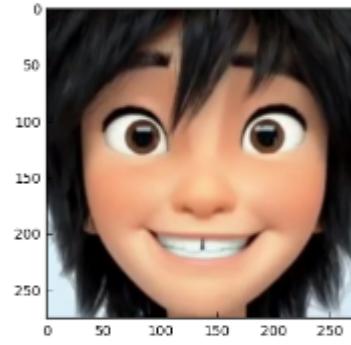
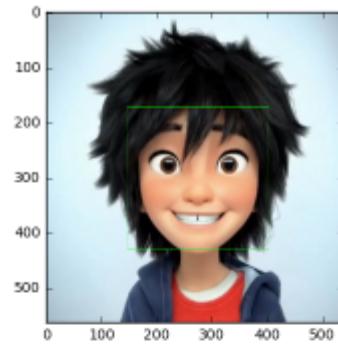


Figura B.7: Test 7



(48, 48)  
1/1 [=====] - 8s  
[3]  
ALEGRE

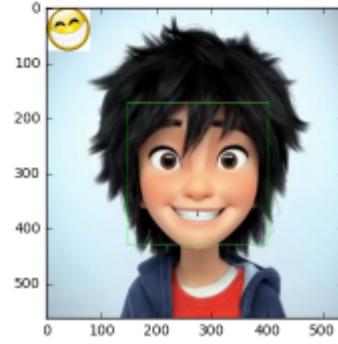


Figura B.8: Test 8

## B.2. Pruebas Fallidas



Figura B.9: Test 9

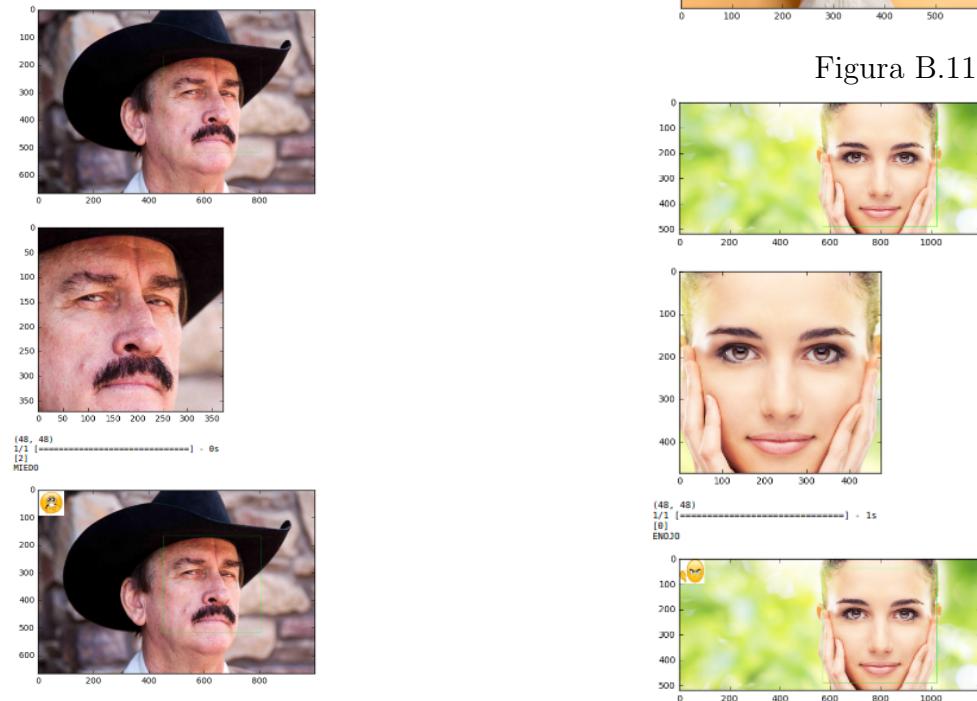


Figura B.10: Test 10

Figura B.11: Test 11

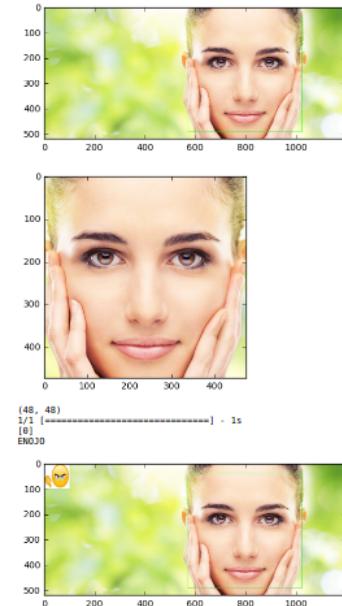


Figura B.12: Test 12

# Apéndice C

## Herramientas

Las herramientas usadas en el desarrollo del presente trabajo, tanto *software* como *hardware* son listados como sigue:

### ■ Software

- Sistema operativo. Linux 14.0
- Lenguaje de programación: Python 2.7
- *Framework deep learning*: Keras

### ■ Hardware

- CPU: Intel XEON 3.4 GHz
- RAM: 8GB

# Apéndice D

## Glosario

- **Convolución:** Operador matemático que transforma 2 funciones en una tercera función.
- **Modelo:** Representación abstracta, conceptual, gráfica, física o matemática, de fenómenos, sistemas o procesos a fin de analizarlos, describirlos, explicarlos, simularlos y predecirlos.
- **Arquitectura:** Técnica y estilo con lo que se diseña, proyecta y construye un modelo.
- **Gradiente:** Derivada parcial de una función respecto a cada variable de ésta.
- **Exabyte:** Unidad de medida de almacenamiento de datos cuyo símbolo es el 'EB' equivalente a  $10^{18}$  bytes.
- **Kaggle:** Plataforma online que ofrece a sus usuarios la opción de participar en distintas competencias cuyo principal tema es el análisis de gran cantidad de datos.
- **Deep Learning (DL)**, es un conjunto de algoritmos en aprendizaje automático que intenta modelar abstracciones de alto nivel en datos usando arquitecturas compuestas de transformaciones no-lineales múltiples.
- **Convolutional Neural Network (CNN)** , tipo de red neuronal artificial donde las neuronas corresponden a campos receptivos de una manera muy similar a las neuronas en la corteza visual primaria de un cerebro biológico.
- **Machine Learning (ML)** , es una rama de la inteligencia artificial cuyo objetivo es desarrollar técnicas que permitan a las computadoras aprender.
- **Red Neuronal Artificial (RNA)**, modelos matemáticos, computacionales, artificiales, ideales de una red neuronal empleados en estadística, psicología cognitiva, e inteligencia artificial.
- **Graphics Processor Unit (GPU)**, es un coprocesador dedicado al procesamiento de gráficos u operaciones de coma flotante, para aligerar la carga de trabajo del procesador central.
- **ImageNet:** Base de datos de imágenes a gran escala con 1.2M de imágenes.

# Apéndice E

## Acrónimos

- **ACM:** Association for Computing Machinery
- **Mm:** Milímetro
- **IEEE:** Institute of Electrical and Electronics Engineers
- **CVPR:** Siglas en inglés de Conference on Computer Vision and Pattern Recognition.
- **VLSI:** Sigla en inglés de Very Large Scale Integration