**TECNOLÓGICO DE ESTUDIOS SUPERIORES DE JOCOTITLÁN**

**RESIDENCIA PROFESIONAL**

ASISTENTE VIRTUAL BASADO EN VISIÓN POR COMPUTADORA

ÁREA DE INVESTIGACIÓN, DIV. INGENIERÍA EN SISTEMAS COMPUTACIONALES

DURACIÓN DEL PROYECTO: 17 DE SEPTIEMBRE 2018 – 28 DE ENERO DE 2019

RESIDENTE: JESÚS SALATIEL BUSTAMANTE ROBLES

ASESOR INTERNO: M. EN T. C ERIKA LÓPEZ GONZÁLEZ

ASESOR EXTERNO: M. EN C. C JUAN CARLOS AMBRIZ POLO

JOCOTITLÁN, MÉXICO ENERO 2019



**ÍNDICE**

[**1** **INTRODUCCIÓN** 6](#_Toc534928985)

[**2** **JUSTIFICACIÓN** 7](#_Toc534928986)

[**3** **OBJETIVOS** 8](#_Toc534928987)

[3.1 OBJETIVO GENERAL 8](#_Toc534928988)

[3.2 OBJETIVO ESPECIFICO 8](#_Toc534928989)

[**4** **CARACTERIZACIÓN DEL ÁREA** 9](#_Toc534928990)

[**5** **PLANTEAMIENTO DEL PROBLEMA** 10](#_Toc534928991)

[**6** **ALCANCES Y LIMITACIONES** 11](#_Toc534928992)

[6.1 ALCANCES 11](#_Toc534928993)

[6.2 LIMITACIONES 11](#_Toc534928994)

[**7** **MARCO TEÓRICO** 12](#_Toc534928995)

[7.1 DEFINICIÓN DE ASISTENTE VIRTUAL 12](#_Toc534928996)

[7.1.1 ANTECEDENTES 12](#_Toc534928997)

[7.1.2 CORTANA 12](#_Toc534928998)

[7.1.3 SIRI 12](#_Toc534928999)

[7.1.4 ALEXA 13](#_Toc534929000)

[7.2 INTELIGENCIA ARTIFICIAL 14](#_Toc534929001)

[7.2.1 VISIÓN ARTIFICIAL 14](#_Toc534929002)

[7.2.2 APRENDIZAJE AUTOMÁTICO 16](#_Toc534929003)

[7.2.3 REDES NEURONALES ARTIFICIALES 17](#_Toc534929004)

[7.2.4 APRENDIZAJE PROFUNDO 19](#_Toc534929005)

[7.2.5 REDES NEURONALES CONVOLUCIONALES 21](#_Toc534929006)

[7.2.6 CLASIFICADORES 23](#_Toc534929007)

[7.2.7 TRANSFERENCIA DEL APRENDIZAJE 24](#_Toc534929008)

[7.2.8 MODELO VGG 25](#_Toc534929009)

[7.3 BASES DE DATOS 26](#_Toc534929010)

[7.3.1 GESTORES DE BASES DE DATOS 26](#_Toc534929011)

[7.4 LENGUAJES DE PROGRAMACIÓN 27](#_Toc534929012)

[7.5 FRAMEWORKS 29](#_Toc534929013)

[**8** **DESARROLLO** 31](#_Toc534929014)

[8.1 ENTORNO DE TRABAJO 31](#_Toc534929015)

[8.2 DEFINICIÓN DE REQUERIMIENTOS 31](#_Toc534929016)

[8.3 ANÁLISIS DE REQUERIMIENTOS 31](#_Toc534929017)

[8.4 CODIFICACIÓN DE ALGORITMOS 32](#_Toc534929018)

[8.5 CONSTRUCCIÓN DEL ASISTENTE VIRTUAL 32](#_Toc534929019)

[8.6 PRUEBAS, DEPURACIÓN DE ALGORITMOS Y AJUSTES DEL ASISTENTE 32](#_Toc534929020)

[**9** **RESULTADOS** 33](#_Toc534929021)

[**10** **CONCLUSIONES Y RECOMENDACIONES** 34](#_Toc534929022)

[**11** **REFERENCIAS** 35](#_Toc534929023)

[**12** **ANEXOS** 40](#_Toc534929024)

**ÍNDICE DE FIGURAS**

[Figura 1: Esquema de neurona artificial [16]. 18](#_Toc534929025)

[Figura 2: Suma ponderada [17] 18](#_Toc534929026)

[Figura 3: Red Neuronal Monocapa [18] 20](#_Toc534929027)

[Figura 4: Red Neuronal Multicapa [18] 20](#_Toc534929028)

[Figura 5: Arquitectura de Red Neuronal Convolucional [19] 21](#_Toc534929029)

[Figura 6: Arquitectura de la Red Neuronal Recurrente 21](#_Toc534929030)

[Figura 7: Arquitectura de una Red Neuronal Convolucional [19]. 22](#_Toc534929031)

[Figura 8: Convolución hace pasar las imágenes de entrada a través de un conjunto de filtros convolucionales, y cada uno de ellos activa determinadas características de las imágenes [21]. 22](#_Toc534929032)

[Figura 9: ReLU permite un entrenamiento más rápido y efectivo al asignar los valores negativos a 0 y mantener los valores positivos [22]. 22](#_Toc534929033)

[Figura 10: Ejemplo de Maxpool con un filtro de 2 x2 y un paso de 2. 23](#_Toc534929034)

[Figura 11: Capa totalmente conectada 23](#_Toc534929035)

[Figura 12: Ecuación que define a la función Softmax 24](#_Toc534929036)

[Figura 13: Frontera entre diferentes clases aplicando el clasificador SVM 24](#_Toc534929037)

[Figura 14: Arquitectura de VGG 25](#_Toc534929038)

[Figura 15: Configuración de la red neuronal. 25](#_Toc534929039)

[Figura 16: Parámetros aprendidos por cada capa. 26](#_Toc534929040)

[Figura 17: Descripción de la metodología. 31](#_Toc534929041)

[Figura 18. Identificación de rostro 33](#_Toc534929042)

**ÍNDICE DE TABLAS**

[Tabla 2: Comparación entre los distintos asistentes virtuales. 13](#_Toc534825935)

[Tabla 3: Funciones de Activación más usuales [17] 19](#_Toc534825936)

[Tabla 4: Comparación entre los distintos gestores de bases de datos. 26](#_Toc534825937)

[Tabla 5: Comparación entre los distintos lenguajes de programación. 28](#_Toc534825938)

[Tabla 6: Comparación entre los distintos Frameworks. 29](#_Toc534825939)

# **INTRODUCCIÓN**

Los avances tecnológicos han tenido gran impacto en la vida cotidiana del ser humano, convivimos con diversos sistemas tecnológicos, la mayor parte del tiempo y cada vez son más las innovaciones que emergen para facilitar la vida. Dentro de las instituciones es de gran relevancia el contar con personal de apoyo como son los asistentes personales o las secretarias, mismos que le permiten al personal superior mejorar su productividad en las jornadas de trabajo, sin embargo, el contar con este tipo de personal no siempre es posible debido al costo que esto conlleva. En este trabajo se describe la implementación de un sistema computacional que funcione como un asistente virtual basado en visión por computadora el cual permita emular las actividades propias de dicho personal. Esta problemática puede ser atendida gracias a los avances de las ciencias computacionales, haciendo uso de nuevas tecnologías emergentes, como son los algoritmos de interacción humana y técnicas de visión artificial. La visión artificial por su parte ayuda a adquirir, procesar y analizar imágenes del mundo real con el fin de producir información numérica que pueda ser tratada por un ordenador. Por otra parte, los asistentes virtuales son herramientas inteligentes que ayudan a los usuarios a buscar información en un conglomerado de recursos digitales todo esto a partir de una interfaz intuitiva para el usuario. Se ha podido integrar un sistema que reconoce a un usuario propietario y detecta a usuarios visitantes los cuales podrán dejar mensajes que posteriormente serán notificados al usuario propietario en cuanto el sistema lo detecte. Finalmente se presentan los resultados preliminares de la investigación y se ponen en discusión para realizar las mejoras pertinentes al sistema.

# **JUSTIFICACIÓN**

La mayoría de las instituciones cuentan con estructuras organizacionales bien definidas donde se describen áreas y puestos que existen dentro de estas. Por lo cual generar nuevos puestos que permitan la contratación de asistentes o secretarias es casi imposible debido al gasto que esto conlleva. Observando esta problemática surge la idea de crear un asistente virtual que apoye a realizar tareas como: capacidad de identificación del jefe inmediato, interacción con personas externas, capacidad de recolección y notificación de mensajes. Dichas tareas pueden ser desarrolladas con el apoyo de tecnología emergente como es la visión artificial además de algoritmos para la interacción humana [3], la fusión de estas tecnologías permite virtualizar las funciones que desempeña un asiste personal generando el mismo impacto al ayudar a incrementar la productividad del usuario propietario.

# **OBJETIVOS**

## OBJETIVO GENERAL

Diseñar e Implementar un Asistente Virtual Basado en Visión por Computadora que permita emular las actividades propias de un asistente personal.

## OBJETIVO ESPECIFICO

* Analizar y entender el reconocimiento e identificación de rostros a partir de imágenes.
* Implementar algoritmos inteligentes que permitan la identificación y reconocimiento de rostros.
* Aplicar algoritmos de adquisición de imágenes que transfieran información a los algoritmos de identificación y reconocimiento de rostros.
* Implementar una interfaz de software que permita el control del sistema.
* Estructurar el asistente virtual a partir de los algoritmos necesarios y la interfaz de software.
* Realizar diversas pruebas al sistema diseñado para analizar resultados obtenidos y retroalimentar el proyecto.
* Documentar el desarrollo del proyecto.

# **CARACTERIZACIÓN DEL ÁREA**

El área donde se desarrolló el proyecto fue el Tecnológico de Estudios Superiores de Jocotitlán una institución educativa encargada de impartir once carreras a nivel licenciatura y un posgrado, ubicada en el norte del Estado de México. El instituto cuenta con catorce líneas de investigación en sus diferentes carreras y un centro de cooperación academia industria (CAI), el proyecto se elaboró en el área de investigación de la división de Ingeniería en Sistemas Computacionales, en el cual cada uno de los docentes de tiempo completo colabora en una línea de investigación, denominada Inteligencia Artificial, la cual el M. en C. C Juan Carlos Ambriz Polo, tiene registrado el proyecto Implementaciones de Visión Artificial a Desarrollos Tecnológicos, en dicha línea se colaboro para el desarrollo de un Asistente Virtual basado en Visión Artificial encargado de reconocer e identificar rostros humanos para la atención personalizada de invitados cuando el administrador se encuentra ausente, además de gestión de notificaciones y comandos de voz.

# **PLANTEAMIENTO DEL PROBLEMA**

Dentro de las diversas instituciones ya sean de índole pública o privada existen múltiples actividades a desarrollar a lo largo de las jornadas de trabajo. En específico existen puestos administrativos u operativos, donde los encargados de dichas áreas presentan una carga excesiva de trabajo y para los cuales es de vital importancia contar con personal tal como secretarias o asistentes personales que apoyen en la gestión y organización del entorno de trabajo. Dentro de las actividades que atienden los asistentes o secretarias se puede considerar desde tomar un recado y trasferir la información, hasta organizar la agenda y hacer notificaciones pertinentes a sus jefes inmediatos.

Por otra parte, se debe considerar que en ocasiones las instituciones no cuentan con los recursos necesarios para contratar a este tipo de personal o incluso en su estructura organizacional no existen dichas figuras**.** De ahí la necesidad de generar un sistema computacional que emule las actividades que realizan este tipo de personal y que no afecte de manera estructural ni económica a las instituciones.

# **ALCANCES Y LIMITACIONES**

## ALCANCES

* Identificación y reconocimiento de caras con un acierto de 95 %.

## LIMITACIONES

* El programa estará instalado en un equipo local por lo cual no podrá ser accedido vía internet.

# **MARCO TEÓRICO**

## DEFINICIÓN DE ASISTENTE VIRTUAL

Un asistente virtual es un programa de computadora que ayuda a los usuarios automatizando y realizando tareas con la mínima interacción hombre-maquina. La interacción que se da entre un asistente virtual y una persona debe ser natural, una persona se comunica usando la voz y el asistente virtual lo procesa y lo interpreta y responde de la misma manera. El asistente personal inteligente tiene sus orígenes en un proyecto de inteligencia militar llamada CALO, por sus siglas en ingles “*Cognitive Assistant that Learns and Organizes*” que significa asistente cognitivo que aprende y organiza [1]. De acuerdo con Kleiner Perkins Caufield & Byers, un ejemplo de tarea que puede ser ejecutado por agentes personales inteligentes incluyen manejo de agenda [2]. Existen elementos que componen un asistente virtual que es dividida por 2 elementos principales.

* INTERFAZ EXTERNA

La interfaz externa representa la forma en que vemos y oímos al asistente virtual ya sea mediante una imagen, animación, un video o una voz. Comúnmente conocido como avatar la interfaz externa aporta al usuario la experiencia real de mantener una conversación con otras personas.

* INTERFAZ INTERNA

La interfaz interna representa toda la programación necesaria para dotar al asistente de una inteligencia virtual capaz de reconocer y procesar el lenguaje empleado por el usuario, así como realizar las acciones solicitadas. Dicha inteligencia se basa en una serie de reglas y algoritmos previamente programados, los avances en este campo tienden a desarrollar complejas inteligencias artificiales que permitan aprender por si sola.

### ANTECEDENTES

La primera herramienta habilitada para realizar el reconocimiento de voz digital fue el Shoebox de IBM, presentado al público en general durante la Feria Mundial de Seattle de 1962 después de su lanzamiento inicial en el mercado en 1961. Esta computadora fue desarrollada casi 20 años antes de la introducción de la primera computadora personal de IBM en 1981, fue capaz de reconocer 16 palabras habladas y los dígitos del 0 al 9. El siguiente hito en el desarrollo de la tecnología de reconocimiento de voz se logró en la década de 1970 en la Universidad Carnegie Mellon en Pittsburgh, Pennsylvania, con un apoyo sustancial del Departamento de Estado de los Estados Unidos. SU herramienta “Harpia” dominaba unas 1000 palabras. Unos diez años después, el mismo grupo de científicos desarrollo un sistema que podía analizar no solo palabras individuales, sino secuencias de palabras completas habilitadas por un Modelo de Markov Oculto [3]. En la década de 1990, la tecnología de reconocimiento de voz digital se convirtió en una característica de la computadora personal. Algunos de los principales asistentes virtuales son:

### CORTANA

Cortana es un asistente virtual creado por Microsoft, puede configurar recordatorios, reconocer la voz natural sin requisito de entrada por teclado y responder preguntas utilizando la información del motor de búsqueda Bing. Cortana se presentó por primera vez en la Conferencia de desarrolladores de Microsoft *BUILD* en San Francisco [4]. El desarrollo de Cortana comenzó en 2009 en el equipo de productos de *Microsoft Speech* con el gerente general de Zig Serafin y el científico jefe Larry Heck.

### SIRI

Siri es un asistente virtual que forma parte de Apple Inc. El asistente utiliza consultas de voz y una interfaz de usuario en lenguaje natural para responder preguntas, hacer recomendaciones y realizar acciones al delegar solicitudes a un conjunto de servicios en Internet [5]. Siri es un derivado de un proyecto desarrollado originalmente por el Centro Internacional de Inteligencia Artificial SRI. Su motor de reconocimiento de voz fue proporcionado por *Nuance Communications*, Siri utiliza tecnologías avanzadas de aprendizaje automático para funcionar. El asistente de voz se lanzó como una aplicación para iOS en febrero de 2010 y fue adquirida por Apple dos meses después. Siri se integró en el iPhone 4S en su lanzamiento en octubre de 2011.

### ALEXA

Amazon Alexa es un asistente virtual desarrollado por Amazon, utilizado por primera vez en los altavoces inteligentes de Amazon Echo. Es capaz de Interactuar con la voz, reproducir música, hacer listas de tareas. Alexa es el servicio de voz basado en la nube de Amazon, Alexa puede crear experiencias de voz naturales que ofrezcan a los usuarios una manera más intuitiva de interactuar con la tecnología que utilizan todos los días [6].

La comparación entre los distintos asistentes virtuales por ventajas y desventajas mas relevantes son mostradas en la Tabla 2.

|  |  |  |
| --- | --- | --- |
| **ASISTENTE VIRTUAL** | **VENTAJAS** | **DESVENTAJAS** |
| CORTANA | * Disponibilidad para Windows 10 * Búsquedas Automáticas en la Web * Registro de preferencias de usuario * Recordatorio de Agenda. | * Necesidad de tener activado la ubicación * Consumo de batería * Actualizaciones gratuitas solo el primer año |
| SIRI | * Facilidad de Uso * Diversidad de funciones * Funciona en todos los dispositivos Apple | * Funciona con una conexión a Internet * Problemas en el Reconocimiento de Voz * Pobre gestión de las notificaciones * Consumo de Batería |
| ALEXA | * Mejor reconocimiento de voz * Equipamiento técnico * Calidad de sonido * Inteligencia lingüística * Multifuncional para Smart Home | * Disponibilidad de idioma solo inglés y alemán * Limitación en la implementación de algoritmos de búsqueda * Problemas en la formulación de frases gramaticales |

Tabla 2: Comparación entre los distintos asistentes virtuales.

## INTELIGENCIA ARTIFICIAL

La inteligencia Artificial es un subcampo de las ciencias computacionales, que busca la creación de máquinas que puedan imitar comportamientos inteligentes similar al humano para la toma de decisiones. Dicho de otra forma, la IA pretende construir sistemas y maquinas que presenten un comportamiento que, si fuera llevado a cabo por una persona, se diría que es inteligente. El aprendizaje, la capacidad de adaptación a entornos cambiantes, la creatividad, etc., son facetas que usualmente se relacionan con el comportamiento inteligente.

Se considera que el origen de la IA se remota a los intentos del hombre desde la antigüedad por incrementar sus potencialidades física e intelectuales, creando artefactos con automatismo y simulando la forma y las habilidades de los seres humanos. La contribución del matemático ingles Alan Turing al mundo de la computación en general, y al de la IA en particular, fue muy considerable. Turing, participo en el diseño de uno de los primeros computadores que existieron, desarrollado para el ejercito inglés, entre los años 40 y 50. Además de aportar los conceptos básicos de la arquitectura secuencial de los computadores en 1950 publicó un artículo que lleva por título “*Computer Machinery and Intelligence*”, que comienza con la pregunta: ¿Pueden las maquinas pensar? En el citado artículo Turing desarrolla la prueba de Turing que sería central en la definición de un elemento artificial que posea “inteligencia”, el famoso “Test de Turing”. En esta prueba se sitúan dos terminales en una sala, uno de ellos conectado a un computador y el otro a un ser humano. Se pretende que el interrogador realice preguntas a través de un puesto conectado a los dos terminales. La prueba es superada si el interrogador no puede discernir cuál de los dos terminales está conectado a un computador y cual está conectado a un ser humano. Si bien el origen inmediato del concepto y de los criterios de desarrollo de la IA se debe a Alan Turing, el apelativo “Inteligencia Artificial” se debe a John McCarthy. McCarthy era uno de los integrantes del “Grupo de Dartmounth” que se reunió en 1956 con fondos de la Fundación Rockerfeller para discutir la posibilidad de construir maquinas que no se limitaran a hacer cálculos prefijados, sino operaciones “inteligentes”.

### VISIÓN ARTIFICIAL

La visión artificial, también conocida como visión por computador es una rama de la Inteligencia Artificial que incluye métodos para adquirir, procesar, analizar y comprender las imágenes del mundo real con el fin de producir información numérica y simbólica para que puedan ser tratados y leídos por una máquina. La visión artificial surge en la década de los 60 con la idea de conectar una cámara de video a un computador; esto implico no solo la captura de imágenes a través de la cámara sino también la comprensión de lo que estas imágenes representaban. Un resultado muy importante de este trabajo y que marco el inicio de la visión artificial, fue un trabajo de Larry Roberts, el creador de ARPA net. En 1961 creo un programa. El “mundo de microbloques”, en el que un robot podía “ver” una estructura de bloques sobre una mesa, analizar su contenido y reproducirla desde otra perspectiva, demostrando esa información visual que había sido mandada al ordenador por una cámara, había sido procesada adecuadamente por él.

COMPONENTES DE VISIÓN ARTIFICIAL

Los sistemas de visión artificial dependen de diversos componentes que trabajan juntos para adquirir, procesar y analizar imágenes. Estos componentes funcionan de manera conjunta con la finalidad de aumentar la calidad de la imagen. Algunos factores para tomar en cuenta para un sistema de visión artificial son: Iluminación, sensores de imagen y el procesamiento de imagen.

#### FUENTES DE ILUMINACIÓN

La iluminación es una de las claves del éxito en los resultados de la visión artificial. Los sistemas de visión artificial crean imágenes a través del análisis de la luz reflejada por un objeto, no del análisis del propio objeto. Una técnica de iluminación implica una fuente de luz y su ubicación con respecto al objeto y la cámara. Una técnica de iluminación particular puede mejorar una imagen de forma que se anule algunas características y se mejoren otras, siluetando una pieza que oscurece los detalles superficiales para permitir la medición de sus bordes, por ejemplo.

##### RETROILUMINACIÓN: La retroiluminación mejora el contorno de un objeto para aplicaciones que solo necesitan medidas externas o de bordes. La retroiluminación ayuda a detectar formas y hace más fiables las medidas dimensionales [7].

##### ILUMINACIÓN DIFUSA AXIAL: La iluminación difusa axial corta la luz en la trayectoria óptica desde el lateral (coaxialmente). Un espejo semitransparente iluminado desde el lateral proyecta la luz hacia abajo sobre la pieza. La pieza refleja la luz hacia la cámara a través del espejo semitransparente, con el resultado de una imagen homogénea e iluminada uniformemente [7].

##### LUZ ESTRUCTURADA: La luz estructurada es la proyección de un patrón de luz (plano, rejilla o forma más compleja) en un ángulo conocido sobre un objeto. Puede ser muy útil para ofrecer inspecciones superficiales independientes del contraste, adquirir información dimensional y calcular volúmenes [7].

#### TIPOS DE CÁMARA

En los diferentes sistemas de visión artificial, la cámara desempeña la función más importante dentro del proceso, debido a que esta actúa como sensor encargado de adquirir la imagen a procesar. Existen dos tipos de cámara, las cuales son:

##### CÁMARAS ANALÓGICAS

Son aquellas cámaras que poseen una señal de salida analógica de video, la cual puede venir acompañada o no, por otras señales de sincronización. La señal de video viene limitada por la anchura de banda y por el ruido análogo que puede proporcionar el cable [8].

##### CÁMARAS DIGITALES

Estas cámaras destacan entre las mas actuales, debido a que poseen sensores sensibles a la luz (foto sensores) y cantidad de componentes electrónicos en su interior, que brindan un mayor desempeño y calidad de la imagen capturada. Los sensores de este tipo de cámaras son fabricados principalmente en dos tecnologías las cuales son:

* Sensor CCD: Un charge coupled device (CCD), convierte las cargas de las celdas de la matriz en voltajes y entrega una señal analógica en la salida, que será posteriormente digitalizada por la cámara. En los sensores CCD, se hace una lectura de cada uno de los valores correspondientes a cada una de las celdas. Entonces, es esta información la que un convertidor analógico-digital traduce en formas de datos. En este caso, la estructura interna del sensor es muy simple, pero se tiene el inconveniente la necesidad de un chip adicional que se encargue del tratamiento de la información proporcionada por el sensor, lo que se traduce en un gasto mayor y equipo más grandes [9].
* Sensor CMOS: En el caso del sensor de tipo CMOS (Complementary Metal Oxide Semiconductor), aquí cada celda es independiente. La diferencia principal es que la digitalización de los pixeles se realiza internamente en unos transistores que lleva cada celda, por lo que todo el trabajo se lleva a cabo dentro del sensor y no se hace necesario un chip externo encargado de esta función [8].

#### PROCESADOR DE VISIÓN

El procesador de visión este compuesto por algoritmos que controlan la imagen y extraen la información necesaria, ejecutan la inspección correspondiente y toman una decisión. Finalmente, la comunicación se suele realizar mediante una señal de Entrada y Salida discreta o información enviada mediante una conexión serial a un dispositivo que registra o usa información.

1. OPENCV

Es una librería de programación de código abierto dirigida principalmente a la visión por computador en tiempo real, desarrollada por la división rusa de Intel en el centro de Nizhni Nóvgorod. La librería OpenCV es multiplataforma, esta fue optimizada para ser usada en procesadores Intel. Uno de los objetivos de OpenCV es proveer una infraestructura de visión por computadora fácil de utilizar que ayude a los programadores a desarrollar aplicaciones sofisticadas de Visión por Computadora rápidamente. La librería contiene aproximadamente 500 funciones que abarcan muchas áreas de Visión por Computadora, incluyendo inspección de productos de fábricas, escaneo médico, seguridad, interfaces de usuario, calibración de cámaras y robótica, porque la visión por computadora y el aprendizaje automático van de la mano. OpenCV también tiene una completa librería de uso general de aprendizaje automático, la cual es muy útil para resolver cualquier problema de esta índole. El proyecto pretende proporcionar un entorno de desarrollo fácil de utilizar y altamente eficiente. Esto se ha logrado, realizando su programación en código C y C++ optimizados, aprovechando además las capacidades que proveen los procesadores multinúcleo. OpenCV puede además utilizar el sistema de primitivas de rendimiento integradas de Intel, un conjunto de rutinas de bajo nivel especificas para procesadores Intel (IPP).

#### APLICACIONES

Las aplicaciones de la visión artificial puedes ser muy variadas y aplicables a distintos tipos de industrias en esta sección solo se mencionarán cuatro aplicaciones de esta.

* DETECCIÓN DE OBJETOS: La detección de objetos es una parte de la visión artificial que estudia como detectar la presencia de objetos en una imagen sobre la base de su apariencia visual.
* APRENDIZAJE AUTOMÁTICO: Las técnicas de aprendizaje automático tienen como objetivo conseguir diferencia automáticamente patrones usando algoritmos matemáticos. Estas técnicas son comúnmente usadas para clasificar imágenes, para tomar decisiones dentro del mundo empresarial, por ejemplo, para decidir que clientes de un bando pueden recibir un préstamo o cuanto han de pagar cada cliente por un seguro dependiendo de sus antecedentes.
* ANÁLISIS DE VIDEO: El análisis de video describe un amplio número de nuevas tecnologías y evoluciones en el campo de la vigilancia con video y la seguridad. Estos cambios están produciendo sistemas de seguridad más efectivos y eficientes.
* VISIÓN 3D: La visión 3D artificial se encarga de proporcionar la capacidad de emular la visión humana a un ordenador. Con dicha capacidad el ordenador podrá generar un modelo tridimensional de un objeto o escena, generalmente a partir de una imagen en 2D.

### APRENDIZAJE AUTOMÁTICO

El aprendizaje automático (Machine Learning) es la rama de la Inteligencia Artificial que tiene como objetivo desarrollar técnicas que permitan a las computadoras aprender. Otra forma mas concreta, se trata de crear algoritmos capaces de generalizar comportamientos y reconocer patrones a partir de una información suministrada en forma de ejemplos. Estos algoritmos son clasificados por:

#### APRENDIZAJE SUPERVISADO: Genera una función que establece una correspondencia entre las entradas y las salidas deseadas del sistema, donde la base de conocimientos del sistema esta formado por ejemplos etiquetados, es decir, ejemplos que sabemos que su clasificación es correcta.

#### APRENDIZAJE NO SUPERVISADO: Todo el proceso del modelado se lleva a cabo sobre un conjunto de ejemplos formado tan solo por entradas al sistema. No se tiene información sobre las categorías de esos ejemplos. Por lo tanto, el sistema tiene que ser capaz de reconocer patrones para poder etiquetar las nuevas entradas.

#### APRENDIZAJE REFORZADO: El algoritmo aprende observando el mundo que le rodea. Su información de entrada la retroalimentación que obtiene del mundo exterior como respuesta a sus acciones. Por lo tanto, el sistema aprende a base de prueba y error, y reforzando aquellas acciones que reciben una respuesta positiva en el mundo.

El aprendizaje automático contiene un conjunto de técnicas de clasificación que ayuda a realizar el aprendizaje de máquina. Estas técnicas son:

* ARBOLES DE DECISIÓN: El árbol de decisión es una estructura de árbol, donde cada nodo representa un atributo a ser probado; las ramas representan la salida de la prueba y los nodos finales (hojas) representan la clasificación. El algoritmo posee dos fases principales: en la primera fase de crecimiento del árbol, el algoritmo inicia con todo el conjunto de datos como nodos raíz. Los datos son divididos en subconjuntos utilizando algún criterio de división. En la segunda fase, etapa de poda del árbol, el árbol totalmente formado se poda para prevenir el exceso de ajuste del árbol a los datos de entrenamiento [10].
* REDES NEURONALES ARTIFICIALES: Las Redes Neuronales Artificiales (RNA) son redes que utilizan aprendizaje supervisado. Las redes neuronales artificiales imitan las neuronas humanas que forman estructuras neuronales altamente interconectadas para realizar tareas complejas de clasificación [11]. Esas conexiones de varias neuronas en estructuras jerárquicas permiten representar limites no lineales de decisiones. Las redes neuronales son una representación simplificada en forma de grafico de una malla de neuronas en el cerebro humano. Los nodos son las unidades de procesamiento, y los vínculos representan las conexiones sinápticas. Para simular la fuerza de las conexiones sinápticas, un peso es asociado con cada conexión entre los nodos de la red neuronal. En cada instante, el estado de un nodo es definido por su nivel de activación. Dependiendo de ese nivel de activación el nodo envía una señal al nodo vecino, la fuerza de esa señal dependerá del peso asociado con esa conexión [12].
* MAQUINA DE VECTORES DE SOPORTE: La máquina de vector de soporte (SVM) es un método que alcanza altos grados de clasificación correcta en diversos tipos de aplicaciones. Es un algoritmo de aprendizaje supervisado, que funciona como un clasificador lineal que separa los datos en dos clases. Básicamente, el algoritmo encuentra entre los diversos hiperplanos que separan esas dos clases el hiperplano ideal que maximiza el margen entre los vectores de soporte. Los vectores de soporte son un subconjunto de datos que solo son los puntos más próximos de las clases, que define la posición del hiperplano ideal [10].

### REDES NEURONALES ARTIFICIALES

Las Redes Neuronales Artificiales (RNA) también llamadas “Sistemas Conexionistas” son un sistema de procesamiento de información inspirado en el funcionamiento del sistema nervioso biológico. Representan un intento de emular el comportamiento del cerebro [13] y aumentar nuestros conocimientos acerca de él. Funcionan como un sistema masivamente paralelo que se ha mostrado eficiente para la resolución inteligente de distintos problemas.

La RNA constan de elementos básicos de proceso, normalmente organizados en capas (basados en las neuronas biológicas) y con gran cantidad de conexiones entre ellos (basadas en la sinapsis de las neuronas).

HISTORIA

En la década de los 40 y principios de los 50, varios investigadores entre los que destacan McCulloch y Pitts Neuman (1943), Householder y Landahl (1945), Kleene (1956), Von Neuman (1956) y Culbertson (1956) elaboraron modelos matemáticos de neuronas y Redes Neuronales. En la década de los 50, varios investigadores, combinaron los resultados obtenidos por los matemáticos, biólogos y desarrollaron modelos de simulación en computadora de neuronas y Redes Neuronales, dando lugar a la forma actualmente más generalizada de trabajar con estos sistemas. Frank Rosenblatt desarrollo el Perceptrón, que fue la primera red neuronal artificial especificada con toda precisión y orientada computacionalmente. En 1962 Rosenblatt público su libro *Principles of Neurodynamics* [14] en el que presento formalmente el Perceptrón como modelo para construir Redes Neuronales Artificiales.

Los perceptrones se aplicaron rápidamente a resolver problemas tales como la predicción climatológica, la interpretación de electrocardiogramas y otros. Tal parecía que se había hallado la clave para comprender el funcionamiento cerebral. Emulando las Redes Neuronales Naturales mediante redes complejas de perceptrones.

Sin embargo, pronto se comprobó que las redes neuronales con una capa de perceptrones eran incapaces de resolver problemas tan simples como la simulación de una compuerta lógica de tipo O exclusivo y, tras una investigación sobre las limitaciones el libro *Perceptrons* [15] donde se hacían publicas estas limitaciones.

LA NEURONA ARTIFICIAL

La neurona artificial es una unidad procesadora con cuatro elementos funcionales: Figura 1



Figura 1: Esquema de neurona artificial [16].

* ENTRADAS: Estas capas reciben la información desde el exterior, que generalmente provienen de otras neuronas y que son atenuadas o amplificadas cada una de ellas con arreglo a un factor de peso Wn que constituye la conectividad entre la neurona fuente de donde provienen y la neurona de destino en cuestión.
* PESOS SINÁPTICOS: Normalmente una neurona recibe muchas y múltiples entradas simultaneas. Cada entrada tiene su propio peso relativo el cual proporciona importancia de la entrada dentro de la función de agregación a la neurona. Estos pesos realizan la misma función que realizan las fuerzas sinápticas de las neuronas biológicas. En ambos casos, algunas entradas son más importantes que otras de manera que tienen mayor efecto sobre el procesamiento de la neurona al combinarse para producir la respuesta neuronal. Los pesos son coeficientes que pueden adaptarse dentro de la red que determinan la intensidad de la señal de entrada registrada por la neurona artificial. Ellos son la medida de la fuerza de una conexión de entrada. Estas fuerzas pueden ser modificadas en respuesta de los ejemplos de entrenamiento de acuerdo con la topología especifica o debido a las reglas de entrenamiento.
* SUMA PONDERADA: Efectúa una suma algebraica ponderada de las señales de entrada, ponderándolas de acuerdo con su peso, aplicando la siguiente expresión, Figura 2:

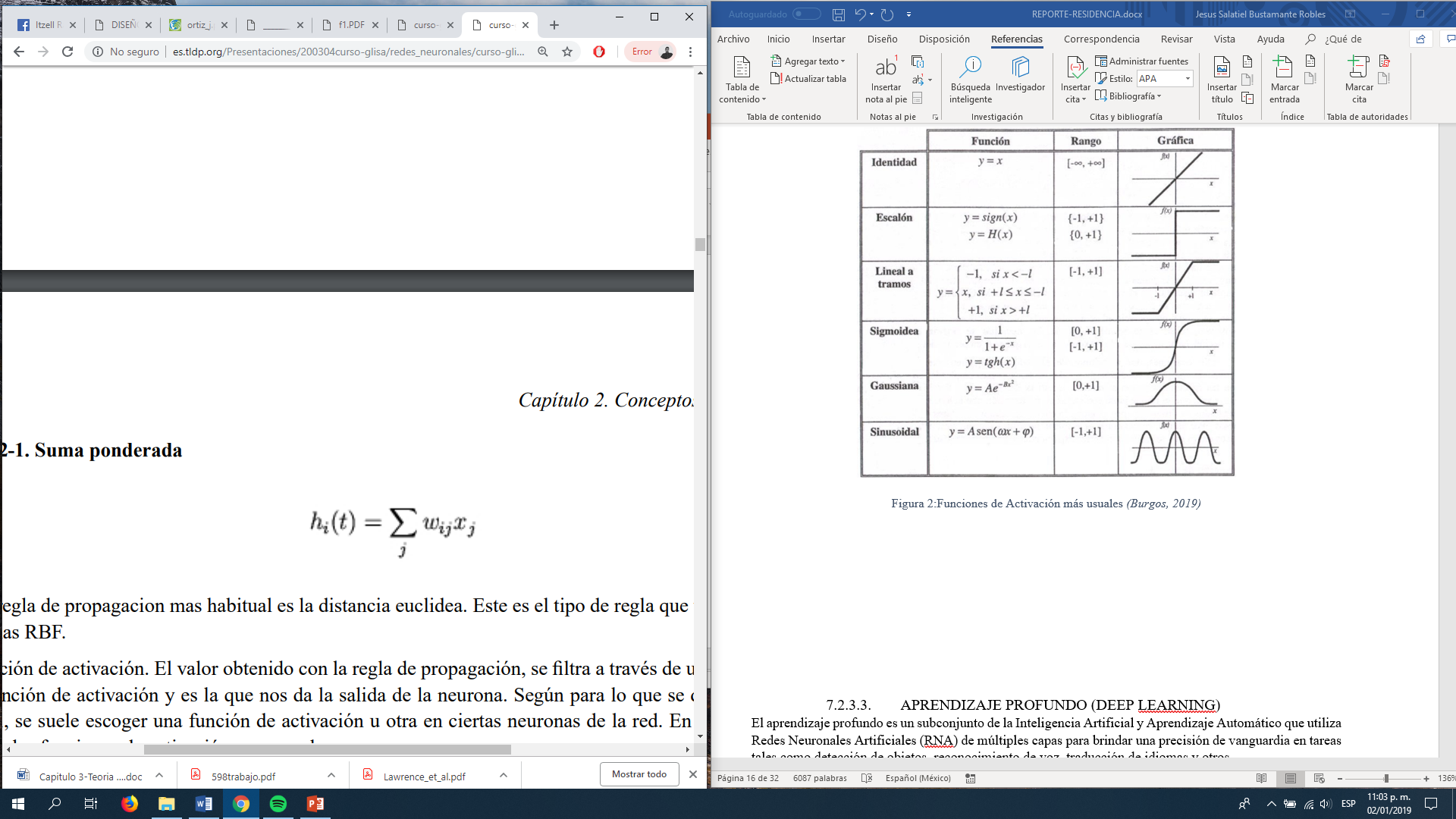


Figura 2: Suma ponderada [17]

* FUNCIÓN DE ACTIVACIÓN: El valor obtenido con la regla de propagación, se filtra a través de una función conocida como función de activación y es la que nos da salida de la neurona. Según para lo que se desee entrenar la red neuronal, se suele escoger una función de activación u otra en ciertas neuronas de la red. En la Tabla 3 se muestran las funciones de activación más usuales

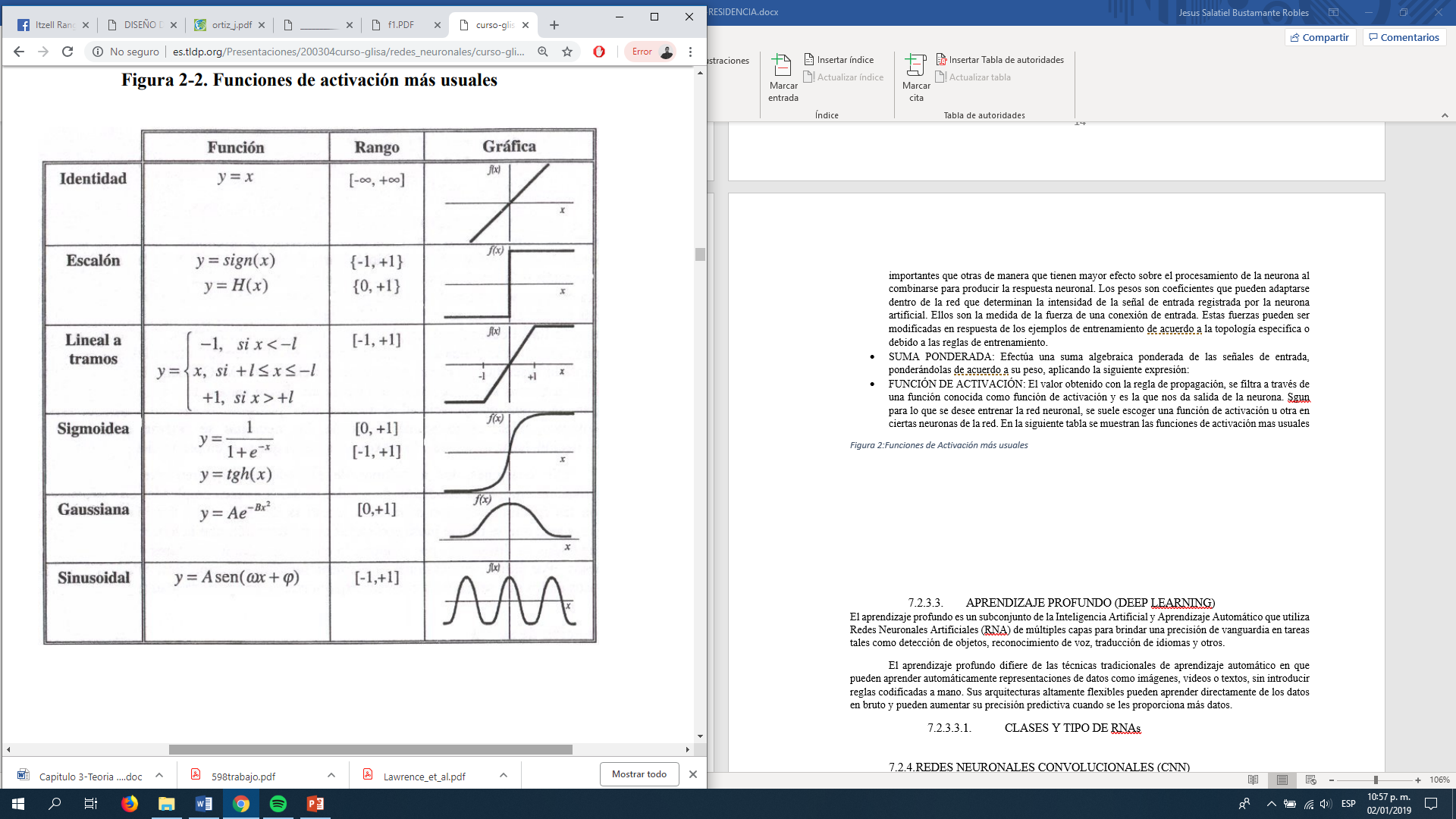


Tabla 3: Funciones de Activación más usuales [17]

En muchas ocasiones la razón para la aplicación de una función de activación distinta de la identidad surge de la necesidad de que las neuronas produzcan una salida acotada. Esto desde un punto de vista de similitud con el sistema biológico, ya que las respuestas de las neuronas biológicas están acotadas en amplitud. Además, cada neurona tiene asociado un numero denominado bías o umbral, que puede verse como un número que indica a partir de que valor del potencial postsináptico la neurona produce una salida significativa.

### APRENDIZAJE PROFUNDO

El aprendizaje profundo es un subconjunto de la Inteligencia Artificial y Aprendizaje Automático que utiliza Redes Neuronales Artificiales (RNA) de múltiples capas para brindar una precisión de vanguardia en tareas tales como detección de objetos, reconocimiento de voz, traducción de idiomas y otros.

El aprendizaje profundo difiere de las técnicas tradicionales de aprendizaje automático en que pueden aprender automáticamente representaciones de datos como imágenes, videos o textos, sin introducir reglas codificadas a mano. Sus arquitecturas altamente flexibles pueden aprender directamente de los datos en bruto y pueden aumentar su precisión predictiva cuando se les proporciona más datos.

#### ARQUITECTURAS DE REDES NEURONALES

La arquitectura de redes neuronales, también denominada diseño de la topología, determina el número de neuronas que tendrá la red, así como su disposición en capas y la conectividad entre las mismas. A partir de un análisis de los datos del problema, se determinan cuantas entradas y salidas tiene la red, así como el número de neuronas y como estas están distribuidas en capas e interconectadas entre sí. Esta etapa es crítica, puesto que la topología de la red determina la capacidad de representatividad de esta, y, por lo tanto, la cantidad de conocimiento que puede albergar. Esta etapa se determina las funciones de activación y trasferencia que se usaran.

Existen diversas clasificaciones de redes neuronales cada una es encargada de resolver problemas específicos aplicados en diversas áreas a continuación se enlistan algunas arquitecturas clasificadas según su topología de red con sus características más relevantes.

##### REDES NEURONALES MONOCAPA: La red neuronal monocapa se corresponde con la red neuronal mas simple, esta compuesta por una capa de neuronas que proyectan las entradas a una capa de neuronas de salida donde se realizan los diferentes cálculos. La arquitectura de esta red neuronal es mostrada en la Figura 3.

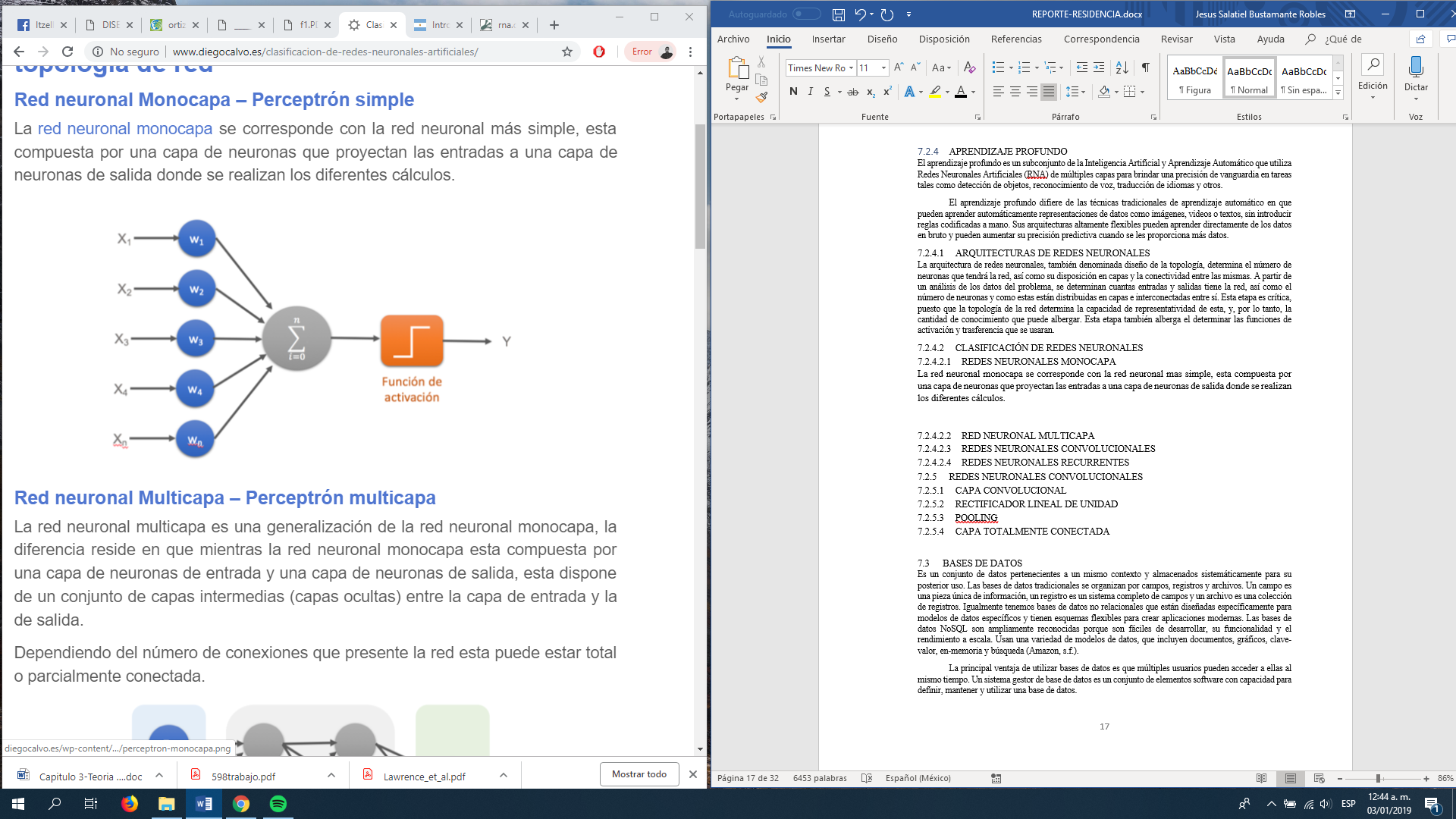


Figura 3: Red Neuronal Monocapa [18]

##### RED NEURONAL MULTICAPA: La red neuronal multicapa es una generalización de la red neuronal monocapa, la diferencia reside en que mientras la red neuronal monocapa esta compuesta por una capa de neuronas de entrada y una capa de neuronas de salida, esta dispone de un conjunto de capas intermedias (capas ocultas) entre la capa de entrada y la capa de salida. Dependiendo del número de conexiones que presente la red esta puede estar total o parcialmente conectada como se muestra en la Figura 4.

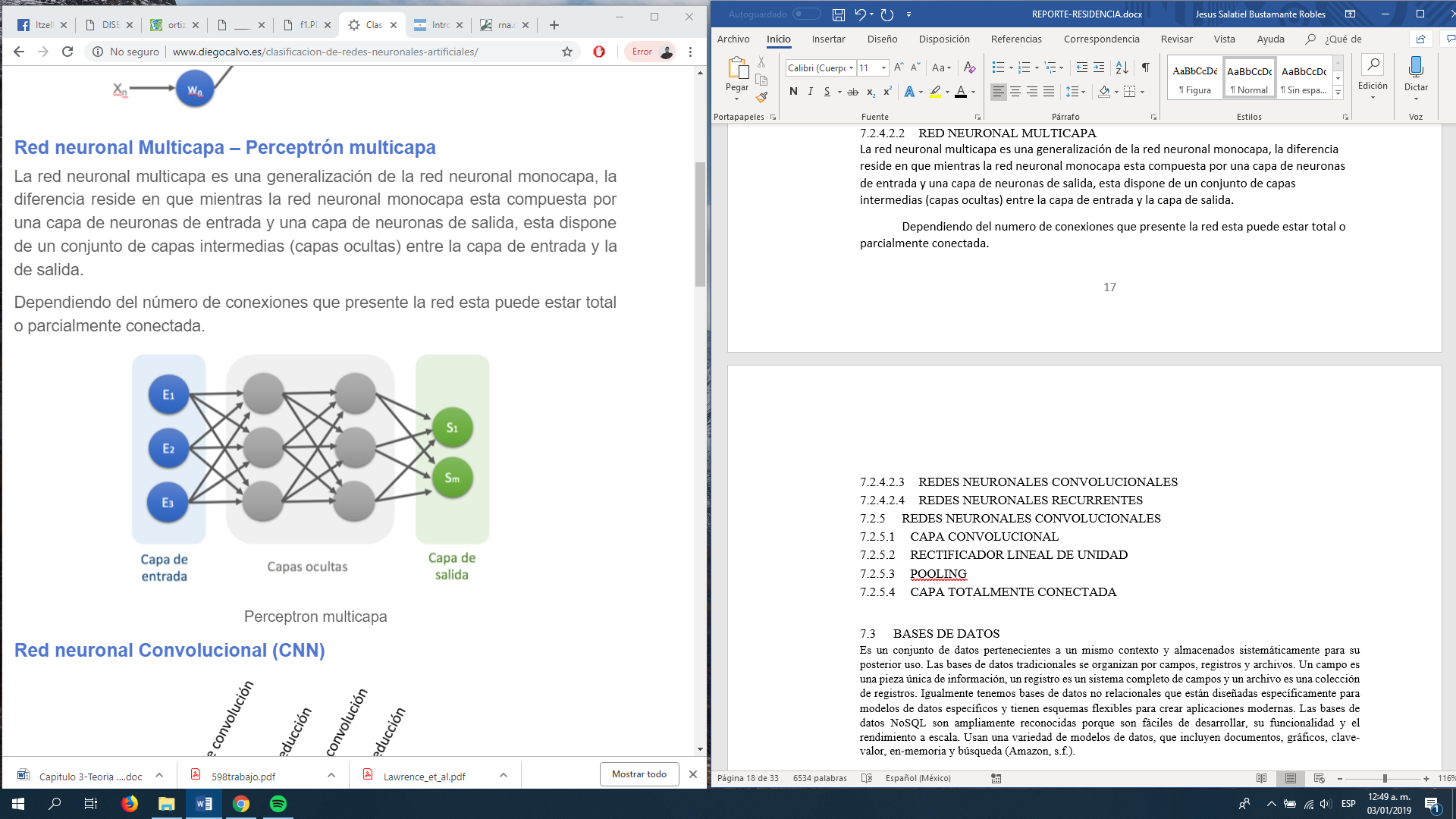


Figura 4: Red Neuronal Multicapa [18]

##### REDES NEURONALES CONVOLUCIONALES: Las redes neuronales convolucionales consisten en múltiples capas de filtros convolucionales de una o mas dimensiones. Como redes de clasificación, al principio se encuentran la fase de extracción de características, compuesta de neuronas convolucionales y de reducción de muestreo, con esto se consigue reducir el numero de neuronas necesarias y la complejidad computacional necesaria para su ejecución. Su arquitectura se muestra en la Figura 5.

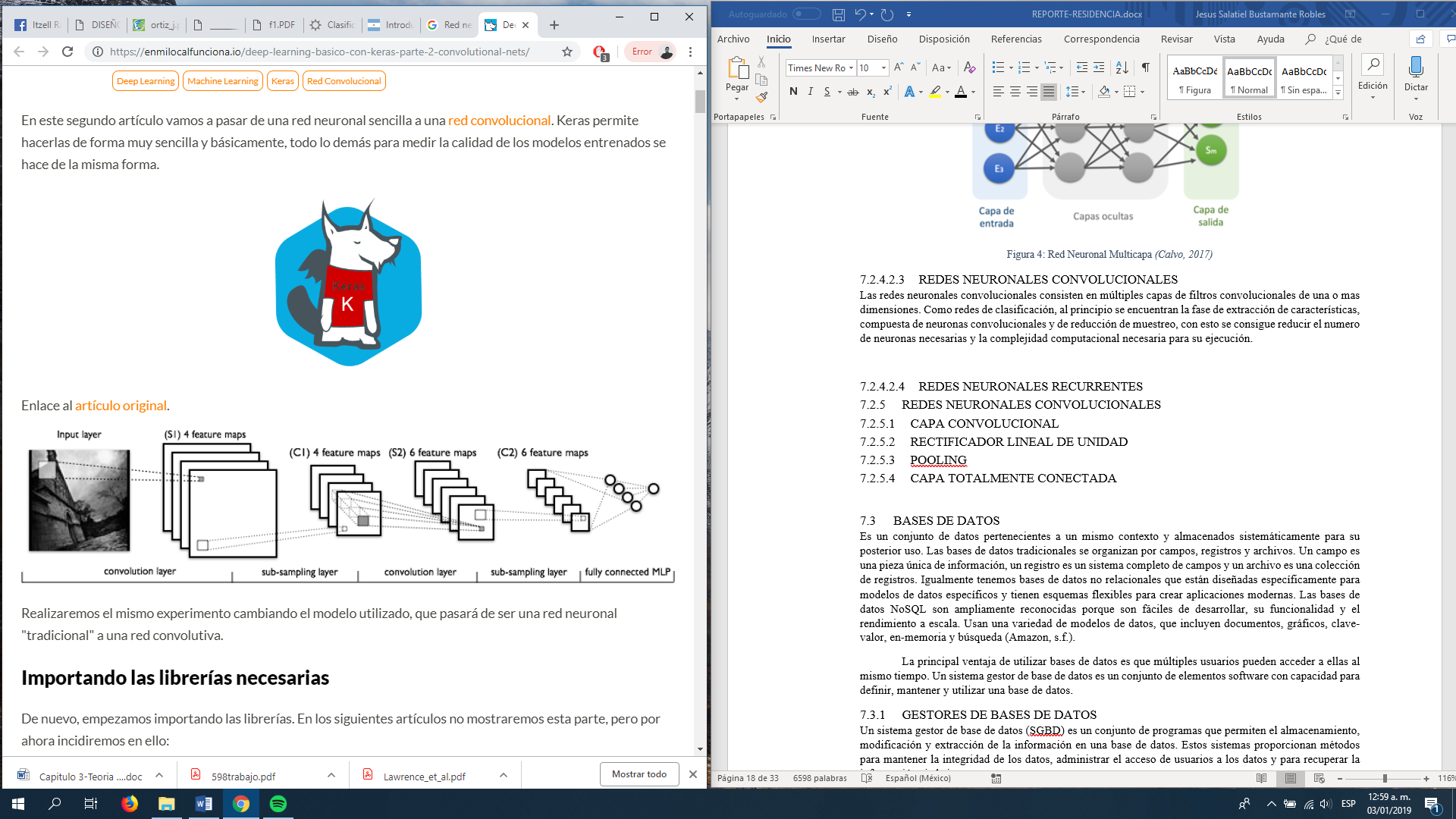


Figura 5: Arquitectura de Red Neuronal Convolucional [19]

##### REDES NEURONALES RECURRENTES: Las redes neuronales recurrentes no tienen una estructura de capas, sino que permiten conexiones arbitrarias entre las neuronas, incluso pudiendo crear ciclos, con esto consigue crear la temporalidad, permitiendo que la red tenga memoria. La mayoría de las aplicaciones de este tipo de red son la identificación y clasificación de patrones secuenciales con distintas posibilidades de ocurrir a través del tiempo, por ejemplo, para traducciones o generación de texto. La arquitectura de esta red es mostrada en la Figura 6.

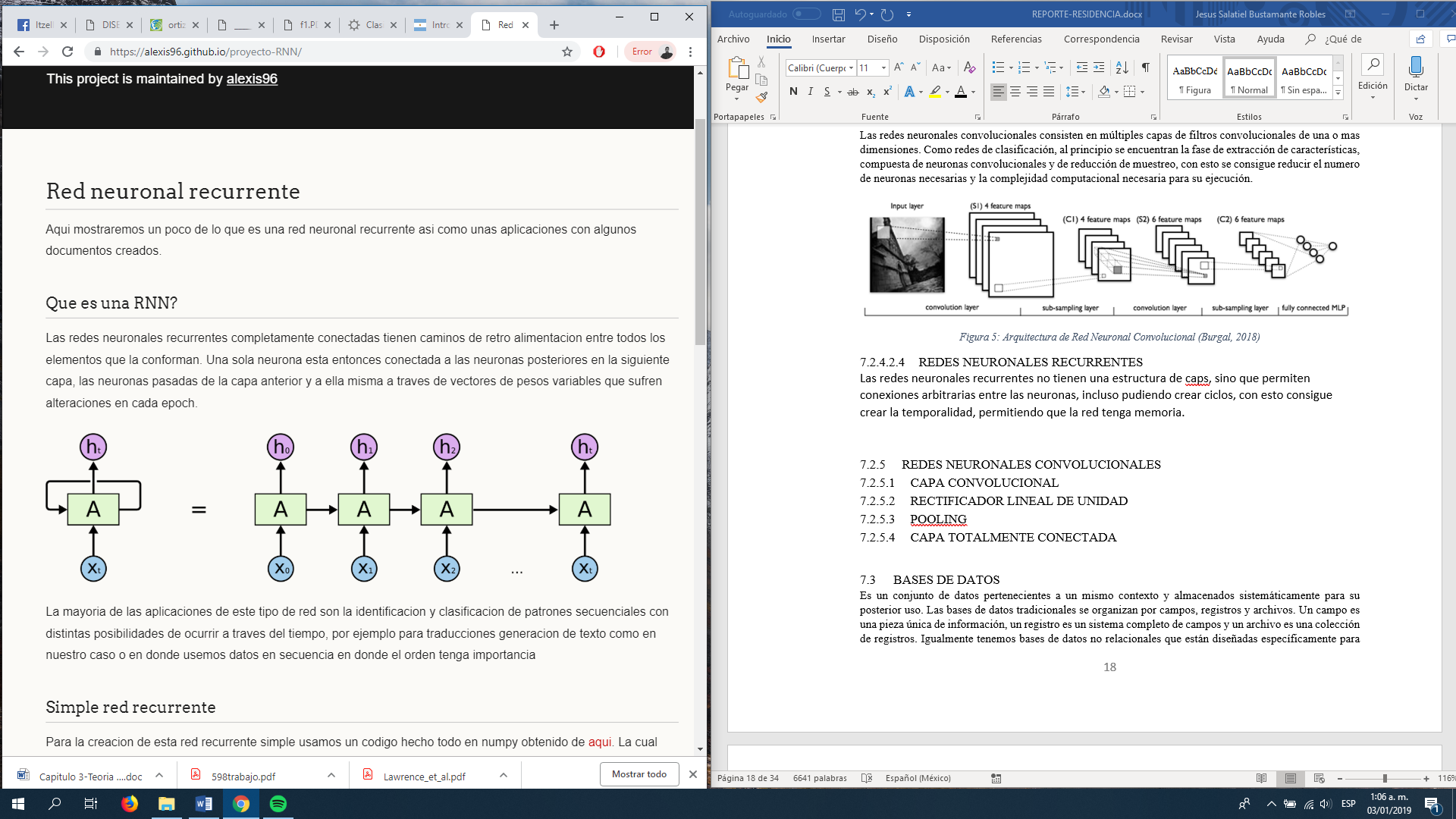


Figura 6: Arquitectura de la Red Neuronal Recurrente

### REDES NEURONALES CONVOLUCIONALES

Una red neuronal convolucional (CNN o ConvNet) es uno de los algoritmos mas populares para Deep learning (aprendizaje profundo), que es un tipo de aprendizaje automático (machine learning) en el que un modelo aprende a realizar tareas de clasificación directamente a partir de imágenes, video, textos o sonidos. Las CNNs son útiles para localizar patrones en imágenes con el objetivo de reconocer objetos, caras y escenas. Aprende a partir de los datos de imágenes, utilizando patrones para clasificar las imágenes y eliminar la necesidad de una extracción manual de características.

Una red neuronal convolucional puede tener decenas o cientos de capas que aprenden a detectar diferentes características de una imagen. Se aplican filtros a cada imagen de entrenamiento con distintas resoluciones, y la salida de cada imagen convolucionada se emplea como entrada para la siguiente capa. Los filtros pueden pasar de características muy simples, tales como el brillo y los bordes, a características más complejas que definen inequívocamente el objeto [20]. La arquitectura utilizada para este tipo de redes neuronales es mostrada en la Figura 7, donde se muestran las operaciones básicas que forman a la red convolucional.

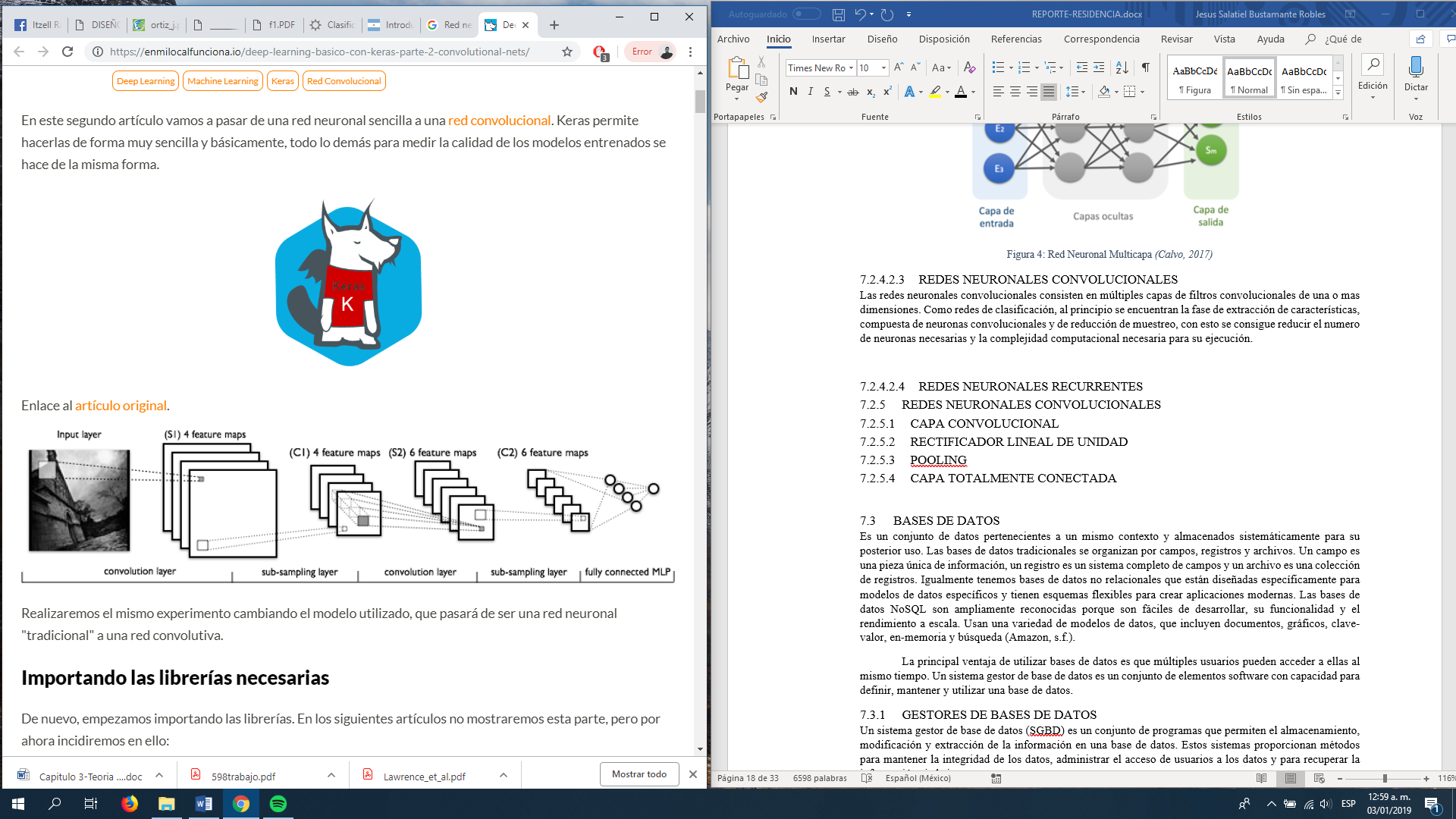


Figura 7: Arquitectura de una Red Neuronal Convolucional [19].

#### CAPA CONVOLUCIONAL: En la convolución se realizan operaciones de productos y sumas entre la capa de partida y los n filtros (kernel) que genera un mapa de características, Figura 8. Las características extraídas corresponden a cada posible ubicación del filtro en la imagen original. La ventaja es que el mismo filtro o neurona sirve para extraer la misma característica en cualquier parte de la entrada, con esto se consigue reducir el numero de conexiones y el número de parámetros a entrenar en comparación con una red multicapa de conexión total.

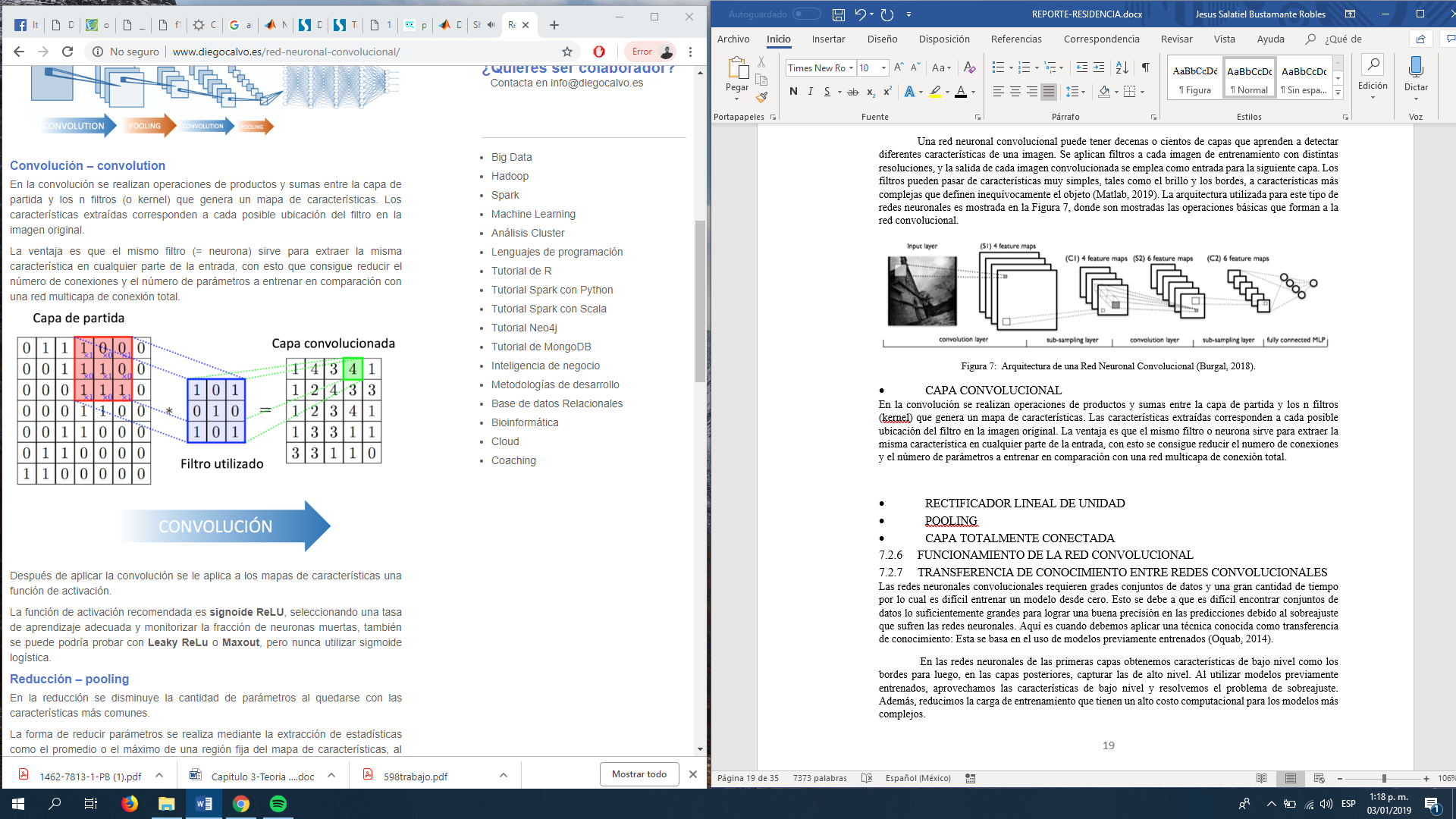


Figura 8: Convolución hace pasar las imágenes de entrada a través de un conjunto de filtros convolucionales, y cada uno de ellos activa determinadas características de las imágenes [21].

#### RECTIFICADOR LINEAL DE UNIDAD: El rectificador lineal de unidad (ReLU) es una función de activación. El propósito de este es introducir la no linealidad en un sistema que básicamente ha estado computando operaciones lineales durante las capas convolucionales. La capa ReLU aplica la función R (z) = max (0, z), Figura 9, a todos los valores en el volumen de entrada. Esta capa simplemente cambia todas las activaciones negativas a 0 sin afectar los campos receptivos de la capa convolucional.

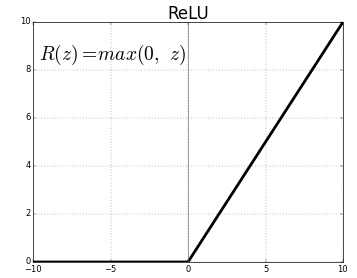


Figura 9: ReLU permite un entrenamiento más rápido y efectivo al asignar los valores negativos a 0 y mantener los valores positivos [22].

#### CAPAS DE AGRUPACIÓN: Capas de agrupación o capa de disminución de resolución (Pooling) es una reducción que disminuye la cantidad de parámetros al quedarse con las características más representativas. La forma de reducir parámetros se realiza mediante la extracción de estadísticas como el promedio o el máximo de una región fija del mapa de características. En esta categoría, también hay varias opciones de capas, siendo Maxpool las más populares. Básicamente, esto requiere un filtro (normalmente de tamaño 2x2) y un desplazamiento de la misma longitud. En la Figura 10 se muestra Maxpool con un tamaño de 2x2 y un paso de 2.

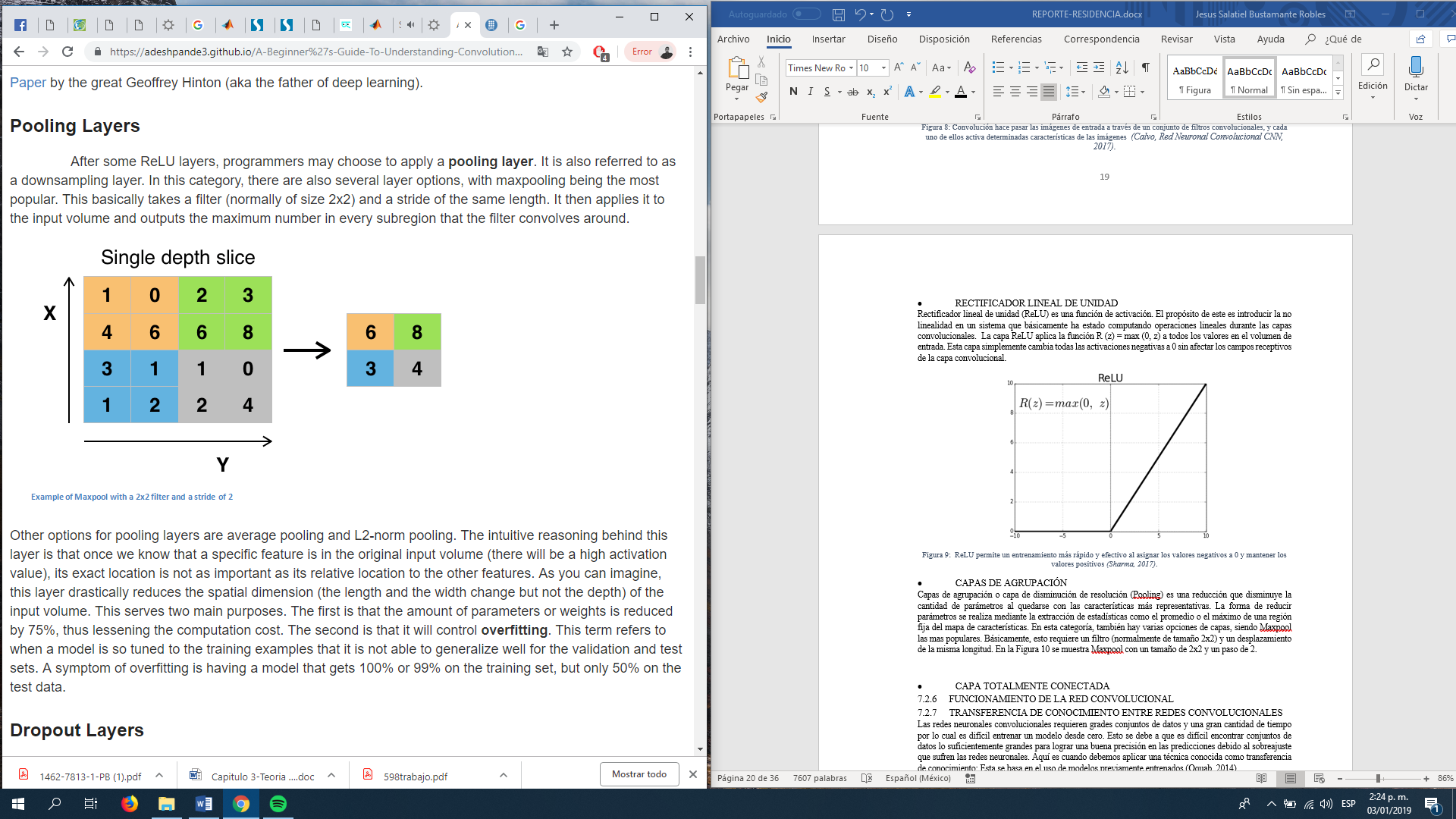


Figura 10: Ejemplo de Maxpool con un filtro de 2 x2 y un paso de 2.

#### CAPA TOTALMENTE CONECTADA: Mediante una convolución 1 x1 este tipo de capas se encargan de generar una salida mono dimensional. Normalmente se posiciona en las ultimas posiciones y se encargan de realizar clasificación o razonamiento de alto nivel puesto que todas las neuronas de esa capa con la anterior están conectadas. En la capa que llamamos capa totalmente conectada (Fully Connected) que se muestra en la Figura 11 [23], aplanamos la matriz en un vector y la introducimos en una capa totalmente conectada como una red neuronal.

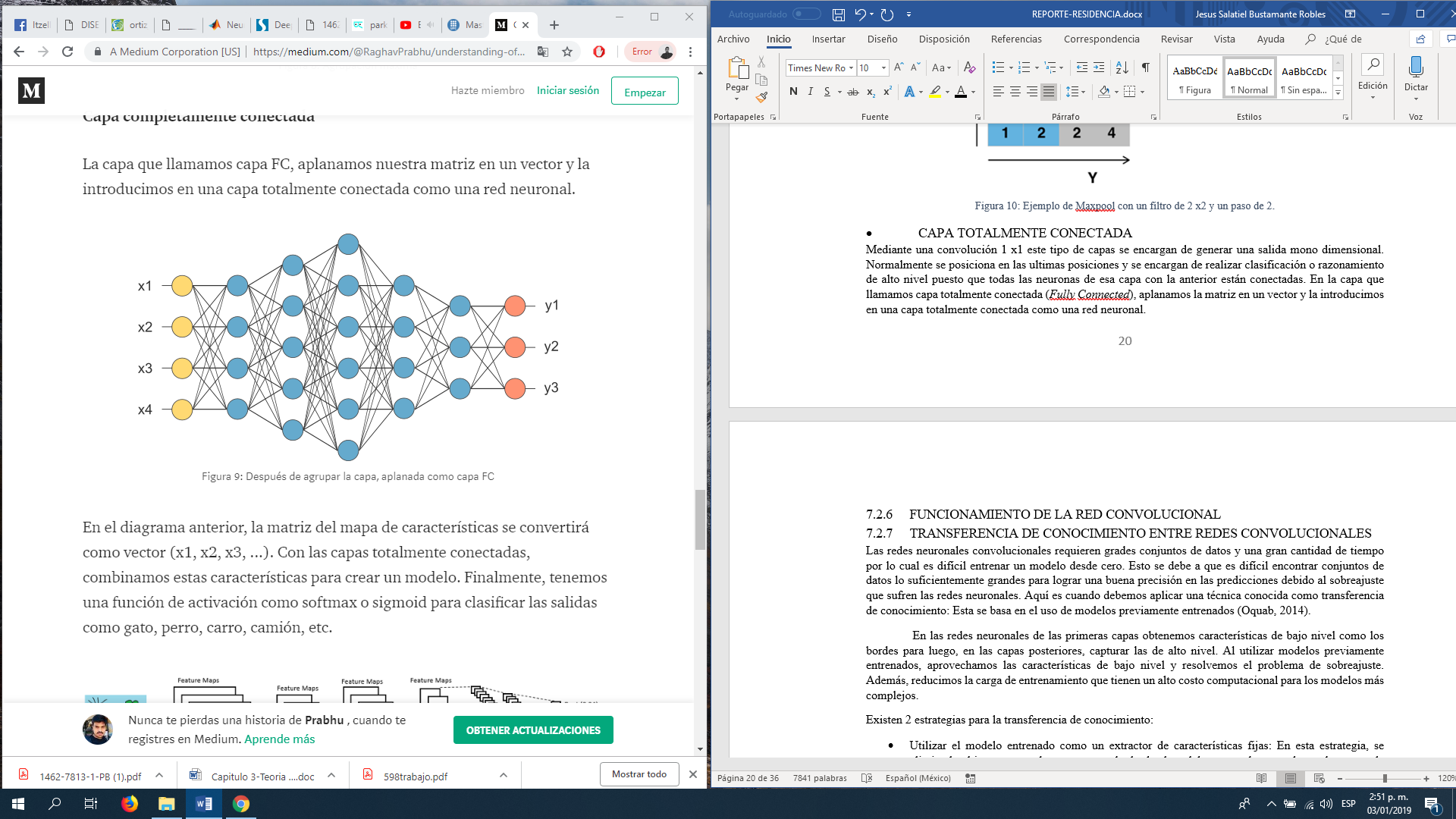


Figura 11: Capa totalmente conectada

En la Figura 11, la matriz del mapa de características se convertirá como vector (x1, x2, x3, ...). Con las capas totalmente conectadas, combinamos estas características para crear el modelo. Finalmente, tenemos una función de activación como softmax o sigmoide para clasificar las salidas.

### CLASIFICADORES

Un clasificador es un algoritmo que, recibiendo como entrada cierta información de un objeto, es capaz de indicar la categoría o clase a que pertenece de entre un numero acotado de clases posibles. En este proyecto se han trabajado con dos opciones mas utilizadas por otros investigadores.

* Softmax: La regresión de Softmax o regresión logística multinomial es una generalización de la regresión logística en el caso en el que queremos manejar varias clases. Esta toma un vector no normalizado y lo normaliza en una distribución de probabilidad. Esta función nos permite mapear un vector de dimensión K (el vector de características) en un vector de probabilidades asignando una probabilidad entre 0 y 1 a cada clase bajo la misma premisa que la suma de las probabilidades de 1, su ecuación que define a la función Softmax mostrada en la figura 12.

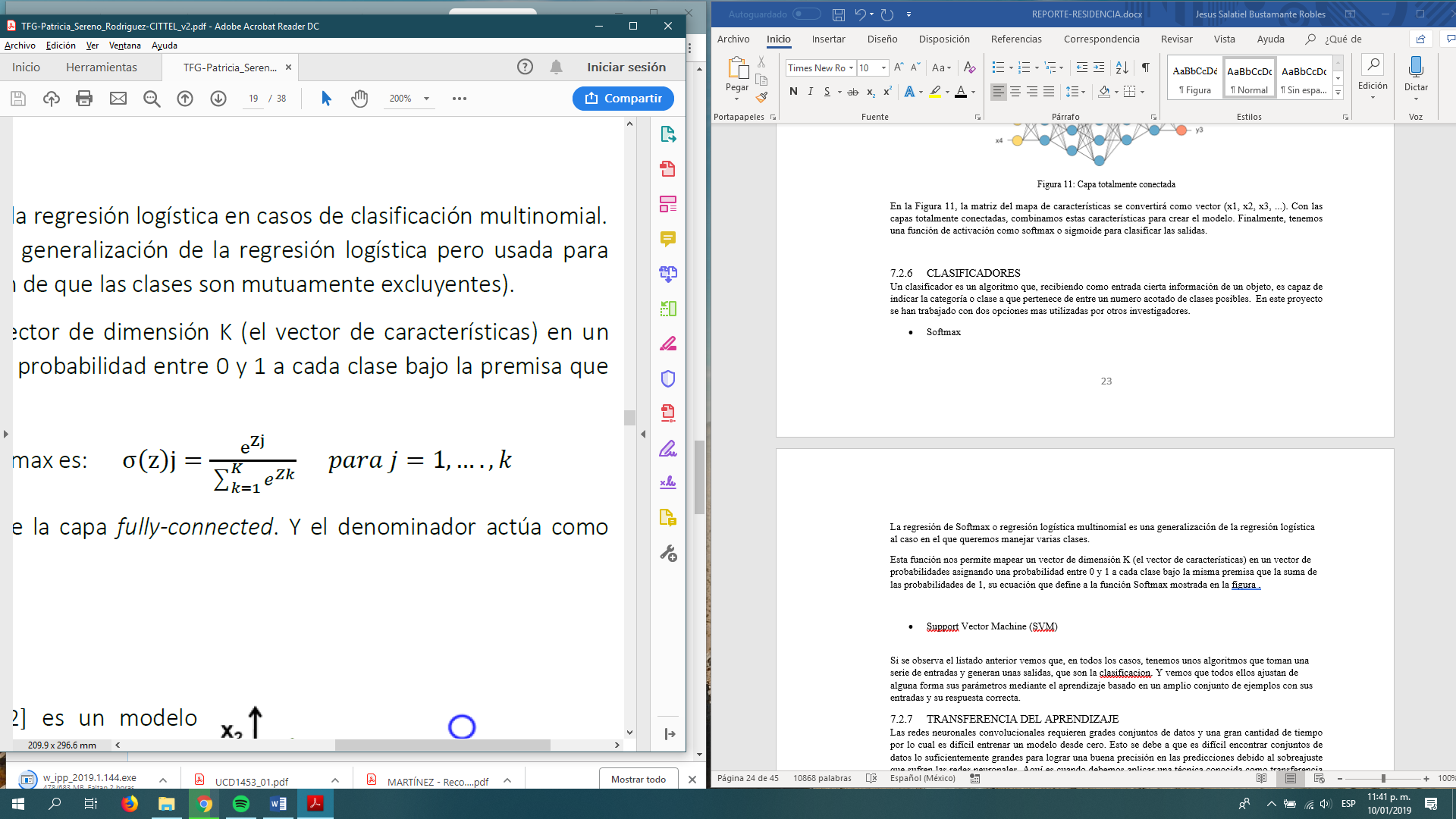


Figura 12: Ecuación que define a la función Softmax

Donde zj es el valor de una neurona de la capa *fully-connected.* Y el denominador actúa como normalizador.

* Support Vector Machine (SVM): La máquina de vectores de soporte (SVM) es un clasificador discriminativo definido formalmente por un hiperplano separador, es decir, es capaz de trazar fronteras entre las diferentes clases de un conjunto de datos sin necesidad de modelar la función de distribución de probabilidad haciendo que se maximice el margen, como es mostrado en la Figura 13. Ya que al maximizar este margen minimiza el error de generalización del clasificador.

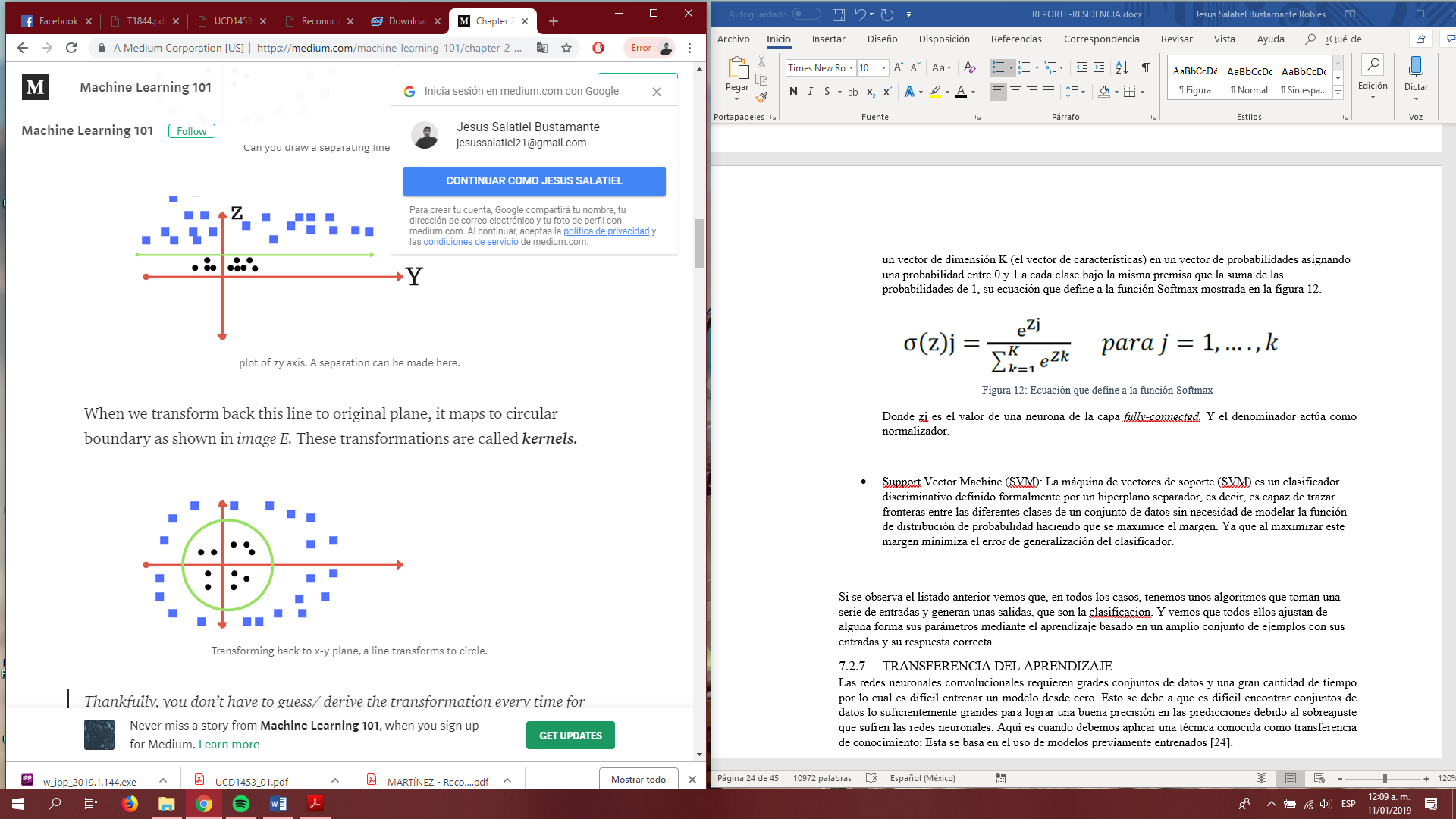


Figura 13: Frontera entre diferentes clases aplicando el clasificador SVM

Si se observa el listado anterior vemos que, en todos los casos, tenemos unos algoritmos que toman una serie de entradas y generan unas salidas, que son la clasificación. Y vemos que todos ellos ajustan de alguna forma sus parámetros mediante el aprendizaje basado en un amplio conjunto de ejemplos con sus entradas y su respuesta correcta.

### TRANSFERENCIA DEL APRENDIZAJE

Las redes neuronales convolucionales requieren grades conjuntos de datos y una gran cantidad de tiempo por lo cual es difícil entrenar un modelo desde cero. Esto se debe a que es difícil encontrar conjuntos de datos lo suficientemente grandes para lograr una buena precisión en las predicciones debido al sobreajuste que sufren las redes neuronales. Aquí es cuando debemos aplicar una técnica conocida como transferencia de conocimiento: Esta se basa en el uso de modelos previamente entrenados [24].

En las redes neuronales de las primeras capas obtenemos características de bajo nivel como los bordes para luego, en las capas posteriores, capturar las de alto nivel. Al utilizar modelos previamente entrenados, aprovechamos las características de bajo nivel y resolvemos el problema de sobreajuste. Además, reducimos la carga de entrenamiento que tienen un alto costo computacional para los modelos más complejos.

Existen 2 estrategias para la transferencia de conocimiento:

* Utilizar el modelo entrenado como un extractor de características fijas: En esta estrategia, se elimina la ultima capa totalmente conectado desde el modelo entrenado, congelamos los pesos de las capas restantes, y formamos un clasificado de aprendizaje automático en la salida de las capas restantes.
* Afinar el modelo entrenado: En esta estrategia, afinar el modelo entrenado en el nuevo conjunto de datos al continuar la propagación hacia atrás. Tenemos la posibilidad de poner a punto toda la red o congelar algunos de sus capas.

Tras realizar algunos ajustes en la red, es posible realizar una tarea nueva, por ejemplo, categorizar solo los perros o los gatos en lugar de 1000 objetos distintos. Esto tiene la ventaja de necesitar mucho menos datos (se procesan miles de imágenes en lugar de millones), de forma que el tiempo de calculo se reduce a horas o minutos [25].

### MODELO VGG

La VGG esta basada en la arquitectura CNN (Convolutional Neural Network) de la VGG-Very-Deep- 16 descrita en [26]. La contiene una estructura de capas convolucionales, de maxpool y capas totalmente conectadas esta arquitectura es mostrada en la Figura 12.

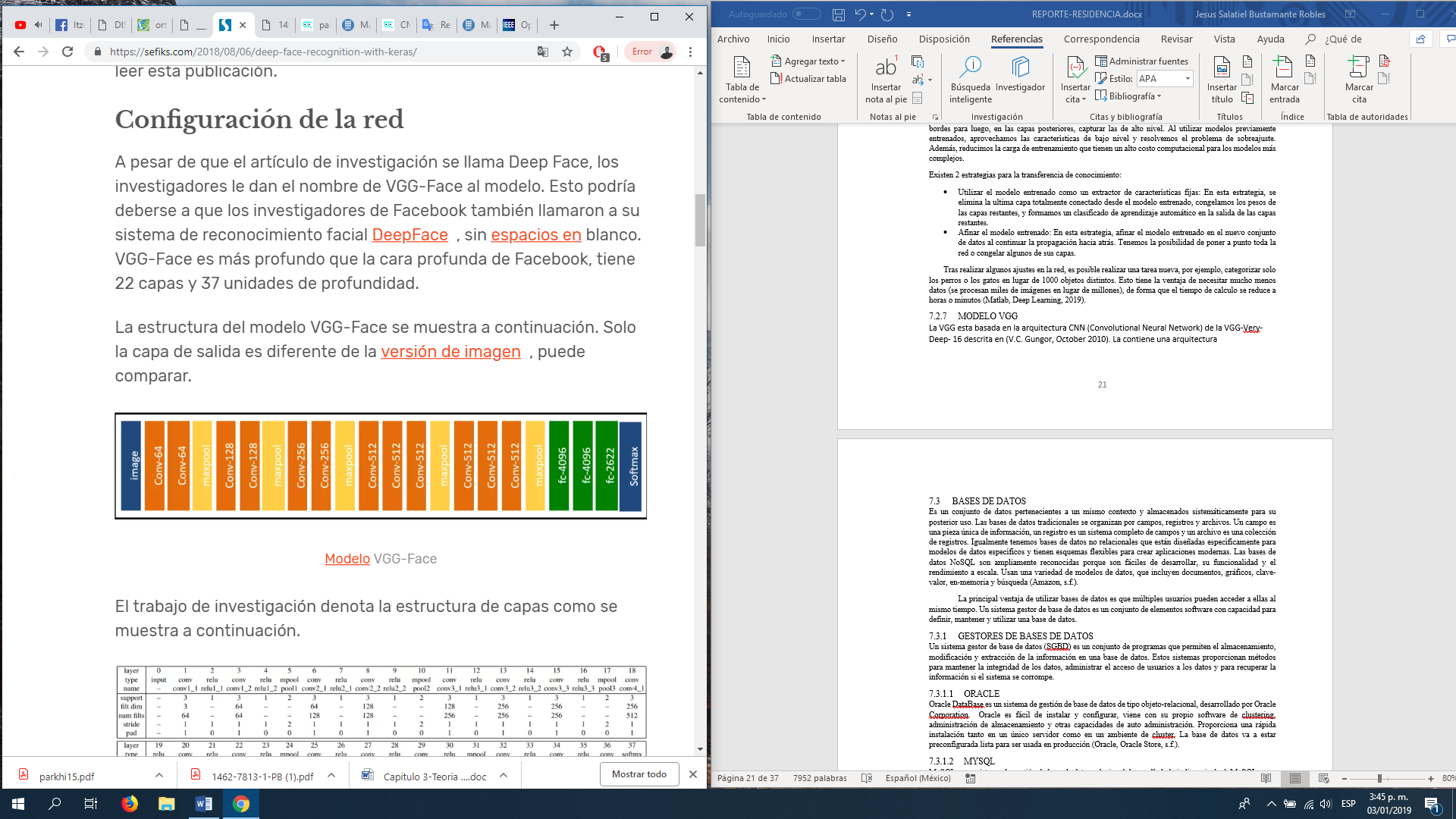


Figura 14: Arquitectura de VGG

Este modelo fue entrenado un conjunto de datos de 2.6 millones de caras. Es comparable al modelo de Google’s FaceNet y superior que Facebook´s DeepFace [27]. En el trabajo de investigación denota las estructuras de capas como se muestra en la Figura 13.

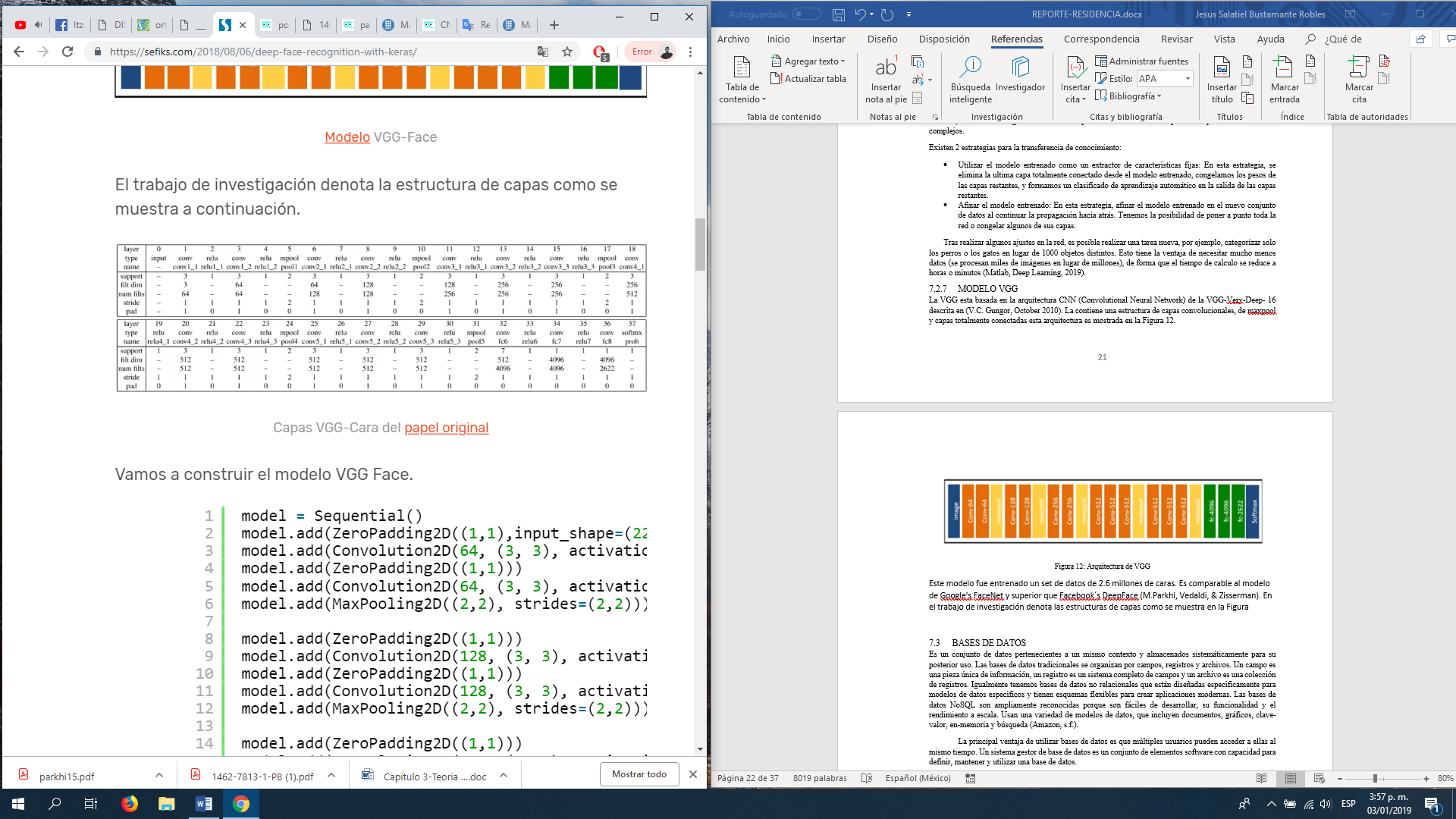


Figura 15: Configuración de la red neuronal.

Este modelo de red esta entrenada solo para identificar a los sujetos en su conjunto de datos de entrenamiento, se puede utilizar como un extractor de características para cualquier otro rostro. La primera capa convolucional utiliza un filtro de profundidad 3 puesto que la entrada de la red es una imagen (224 x 224 pixeles) con codificación RGB. En las otras capas esta profundidad viene establecida por la profundidad anterior. Tenemos 5 capas de maxpool que reducirán la dimensión de la imagen a una de 7 x 7, cada capa tiene un factor de paso de 2. En la salida del último pooling obtendremos una salida con dimensión 7 x 7 x 512. Esta salida se conecta a dos capas totalmente conectadas con dimensión de 4096 neuronas. La salida tendrá una dimensión de 2622 y estará conectado con el clasificador, en este caso softmax, como es mostrado en la, Figura 14, donde se proporcionan detalles adicionales sobre las capas de la CNN. La columna de Dimensiones representa la anchura, altura y profundidad de cada capa. La columna de parámetros muestra el número de parámetros aprendidos [28]*.*

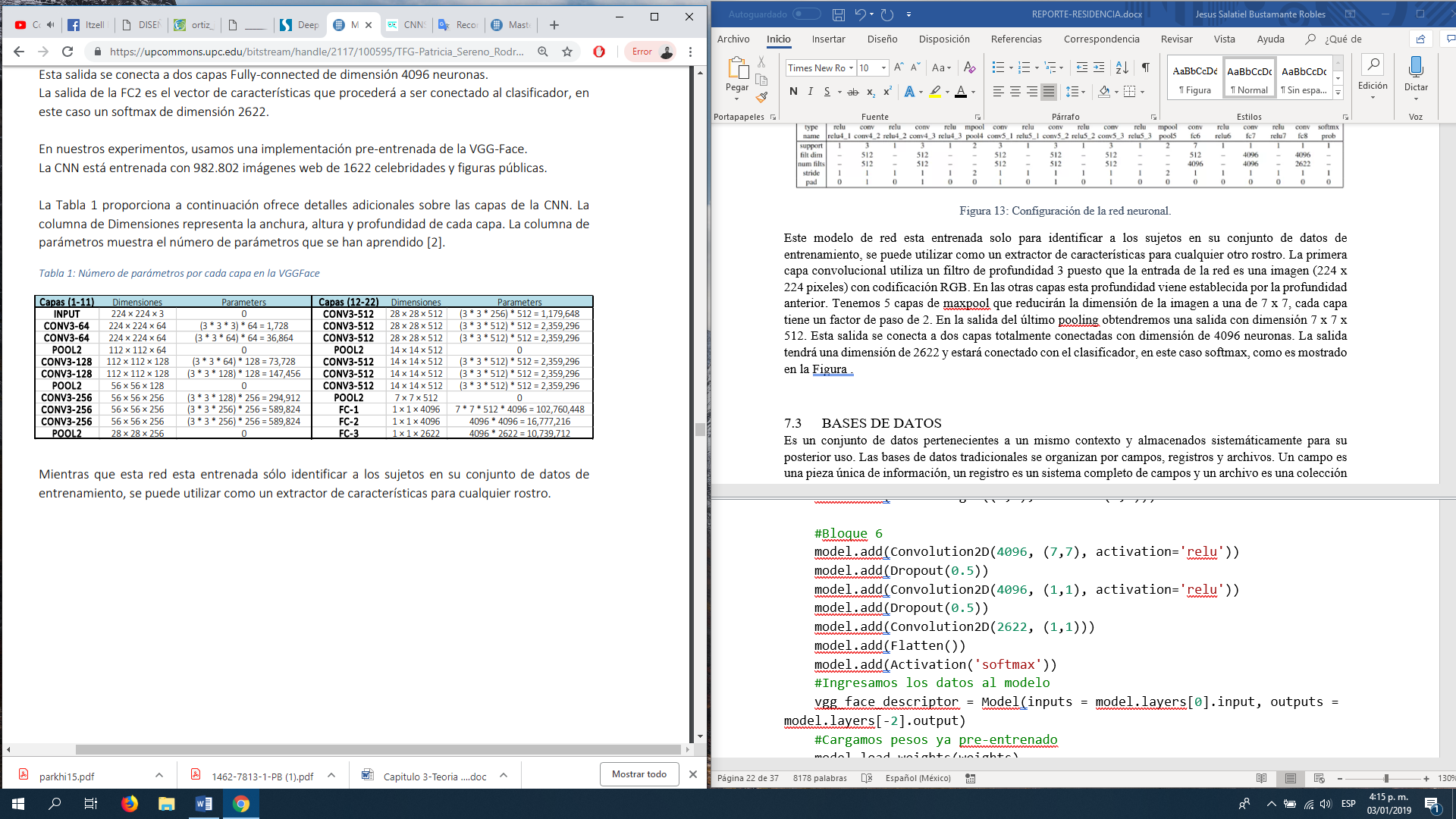


Figura 16: Parámetros aprendidos por cada capa.

## BASES DE DATOS

Es un conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso. Las bases de datos tradicionales se organizan por campos, registros y archivos. Un campo es una pieza única de información, un registro es un sistema completo de campos y un archivo es una colección de registros. Igualmente tenemos bases de datos no relacionales que están diseñadas específicamente para modelos de datos específicos y tienen esquemas flexibles para crear aplicaciones modernas. Las bases de datos NoSQL son ampliamente reconocidas porque son fáciles de desarrollar, su funcionalidad y el rendimiento a escala. Usan una variedad de modelos de datos, que incluyen documentos, gráficos, clave-valor, en-memoria y búsqueda [29].

La principal ventaja de utilizar bases de datos es que múltiples usuarios pueden acceder a ellas al mismo tiempo. Un sistema gestor de base de datos es un conjunto de elementos software con capacidad para definir, mantener y utilizar una base de datos.

### GESTORES DE BASES DE DATOS

Un sistema gestor de base de datos (SGBD) es un conjunto de programas que permiten el almacenamiento, modificación y extracción de la información en una base de datos. Estos sistemas proporcionan métodos para mantener la integridad de los datos, administrar el acceso de usuarios a los datos y para recuperar la información si el sistema se corrompe.

#### ORACLE: Oracle DataBase es un sistema de gestión de base de datos de tipo objeto-relacional, desarrollado por Oracle Corporation. Oracle es fácil de instalar y configurar, viene con su propio software de clustering, administración de almacenamiento y otras capacidades de auto administración. Proporciona una rápida instalación tanto en un único servidor como en un ambiente de cluster. La base de datos va a estar preconfigurada lista para ser usada en producción [30].

#### MYSQL: Es un sistema de gestión de base de datos relacional desarrollado bajo licencia dual. MySQL no es más que una aplicación que permite gestionar archivos llamados de bases de datos. MySQL como base de datos relacional, utiliza múltiples tablas para almacenar y organizar la información. Fue escrito en C y C++ y destaca por su gran adaptación a diferentes entornos de desarrollo, permitiendo su interactuación con los lenguajes de programación más utilizados como PHP, Perl y Java y su integración en distintos sistemas operativos [31].

#### MONGODB: Es una base de datos de documentos de código abierto que proporciona alto rendimiento, alta disponibilidad y escalado automático. Un registro en MongoDB es un documento, que es una estructura de datos compuesta por pares de campos y valores. Los documentos de MongoDB son similares a los objetos JSON. Los valores de los campos pueden incluir otros documentos, matrices y matrices de documentos [32].

La comparación entre los distintos gestores bases de datos por ventajas y desventajas mas relevantes son mostradas en la Tabla 4.

|  |  |  |
| --- | --- | --- |
| **GESTOR DE BASE DE DATOS** | **VENTAJAS** | **DESVENTAJAS** |
| ORACLE | * Buen motor de base de datos objeto-relacional * Multiplataforma * Lenguaje PL/SQL * Uso de particiones | * Uso de licencias de alto costo * Alto coste de capacitación |
| MYSQL | * Open Source (Código abierto) * Velocidad al realizar las operaciones * Rendimiento * Bajo costo de requerimientos de hardware | * No soporta integridad relacional * No tiene transacciones en aplicaciones web |
| MONGODB | * Validación de Documentos * Motores de almacenamiento integrado * Menor tiempo de recuperación ante fallas * Modelo de datos flexible | * No es una solución adecuada para aplicaciones con transacciones complejas * Aun es tecnología joven |

Tabla 4: Comparación entre los distintos gestores de bases de datos.

## LENGUAJES DE PROGRAMACIÓN

Los lenguajes de programación son una forma de interpretación de un lenguaje formal que especifica una serie de instrucciones para que una computadora produzca ciertas funciones previamente establecidas. Estos lenguajes son códigos estructurados que cobran sentido en determinados contextos de uso. Son formados por un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones.

* PYTHON: Es un lenguaje de programación interpretado, de alto nivel, de propósito general. Tiene una filosofía de diseño que enfatiza la legibilidad del código, en particular al usar espacios en blanco significativos. Python cuenta con un sistema de tipo dinámico y gestión automática de memoria. Es compatible con múltiples paradigmas de programación, incluyendo orientado a objetos, imperativo, funcional y de procedimiento, y tiene biblioteca estándar grande y completa [33].
* R: Es un lenguaje de programación y un entorno de software libre para la computación estadística y gráficos. R y sus bibliotecas implementan una amplia variedad de técnicas estadísticas, que incluyen modelos lineales y no lineales, pruebas estadísticas clásicas, análisis de series de tiempo, clasificación, agrupamiento y otros. Muchas de las funciones estándar de R están escritas en R, lo que facilita a los usuarios seguir las elecciones algorítmicas realizadas. Para tareas intensivas en computación, los códigos C, C++ y Fortran se pueden vincular y llamar en tiempo de ejecución. Los usuarios avanzados pueden escribir C, C++, Java. NET o código Python para manipular objetos R directamente [34].
* JAVA: Es un lenguaje de programación de propósito general que es concurrente, basado en clases, orientado a objetos y diseñado específicamente para tener la menor cantidad de dependencias de implementación posibles. Las aplicaciones Java generalmente se compilan a un código de bytes que pueden ejecutarse en cualquier máquina virtual Java (JVM) independientemente de la arquitectura de la computadora [35]. Java tiene una biblioteca para el aprendizaje automático llamado “Java-ML” es un marco de trabajo que proporciona varios algoritmos de aprendizaje automático.
* C++: Es un lenguaje de programación de propósito general. Tiene funciones imperativas, orientado a objetos, al mismo tiempo que proporciona instalaciones para la manipulación de memoria de bajo nivel. Fue diseñado con un sesgo hacia la programación del sistema y sistemas integrados, con recursos limitados y grandes, con el rendimiento, la eficiencia y la flexibilidad de uso como aspectos destacados del diseño.

La comparación entre los distintos lenguajes de programación por ventajas y desventajas mas relevantes son expuestas en la Tabla 5.

|  |  |  |
| --- | --- | --- |
| **LENGUAJE** | **VENTAJAS** | **DESVENTAJAS** |
| PYTHON | * Ofrece varios paquetes para ciencia de datos * Fácil de usar * Interacción con servicios en la nube * Código Abierto * Legibilidad del código | * Lenguaje interpretado |
| R | * Ofrece varios paquetes para ciencia de datos * Interacción con servicios en la nube * Velocidad para realizar análisis estadísticos * Código Abierto | * Lenguaje de programación lento |
| JAVA | * Multiplataforma * Orientado a Objetos * Librerías Estándar para *Machine Learning* * Manejo Automático de Memoria * Gratuito | * Lentitud a la hora de ejecutar aplicaciones * Requiere de un interprete * Algunas herramientas tienen un costo |
| C++ | * Orientado a Objetos * Multiplataforma * Robusto y Versátil * Flexible al programar con múltiples estilos | * Uso de DLLs (Librerías Dinámicas) * Manejo de punteros y memoria |

Tabla 5: Comparación entre los distintos lenguajes de programación.

## FRAMEWORKS

Los marcos de aprendizaje profundo ofrecen componentes básicos para diseñar, capacitar y validar redes neuronales profundas, a través de una interfaz de programación de alto nivel [36]. Dado que el aprendizaje profundo es la clave para ejecutar tareas de mayor nivel de sofisticación, construirlas e implementarlas con éxito demuestra ser un gran desafío para los científicos de datos e ingenieros de datos de todo el mundo. Hoy en día, contamos con una gran variedad de marcos de trabajo (Framework) que nos permiten desarrollar herramientas que pueden ofrecer un mejor nivel de abstracción junto con la simplificación de los desafíos de programación difíciles.

* TENSORFLOW: Es una biblioteca de software de código abierto que es utilizada para realizar cálculos numéricos mediante diagramas de flujo de datos. Los nodos de los diagramas representan operaciones matemáticas y las aristas reflejas las matrices de datos multidimensionales (tensores). En su origen, TensorFlow fue fruto del trabajo de investigadores e ingenieros de Google Brain Team que formaban parte de la organización de investigación del aprendizaje automático de Google [37].
* CAFFE: Es un marco de aprendizaje profundo fue originado por la Universidad de California, Berkeley en 2014, y ha dado lugar a nuevos marcos de trabajo como Caffe2 de Facebook. Tiene una arquitectura expresiva fomenta la explicación y la innovación. Los modelos y la optimización están definidos por la configuración sin codificación [38].
* PYTORCH: Es un paquete de computación científica basado en Python que utiliza el poder de las unidades de procesamiento de gráficos. También es una de las plataformas de investigación de aprendizaje profundo preferidas, diseñadas para proporcionar la máxima flexibilidad y velocidad. Es conocido por proporcionar dos de las características más de alto nivel a saber, los cálculos de tensor con un fuerte soporte de aceleración de GPU y la creación de redes neuronales profundas. [39].
* KERAS: Es una biblioteca de red neuronal de código abierto escrita en Python. Es capaz de ejecutarse sobre TensorFlow, Microsoft Cognitive y Theano. Diseñado para permitir una experimentación rápida con redes neuronales profundas, se enfoca en ser fácil de usar, modular y extensible. Fue desarrollado como parte del esfuerzo de investigación del proyecto ONEIROS (Sistema operativo de robot inteligente neuro electrónico abierto). En 2017 fue apoyado por el equipo de Google Tensorflow para ser una interfaz en lugar de un marco de aprendizaje profundo [40].

Comparación entre los distintos Frameworks expuesta por sus principales características y complejidad de uso (Curva de aprendizaje), son mostrados en la Tabla 6.

|  |  |  |
| --- | --- | --- |
| **FRAMEWORK** | **CARACTERÍSTICAS** | **COMPLEJIDAD DE USO** |
| TENSORFLOW | 1. Rendimiento al trabajar con GPU, CPU y TPU. 2. Flexible al ofrecer APIs de alto nivel que facilitan el desarrollo de modelos. 3. Código Libre 4. Amplia documentación 5. Soporte de una comunidad. 6. API de Python, C++, Java, R 7. Optimizado para tarjetas gráficas NVidia. | Alta |
| CAFFE | 1. Optimización entre CPU y GPU 2. Velocidad de procesamiento. 3. Comunidad 4. Soporte de NVidia 5. Modularidad de componentes 6. APIs de Python, Matlab. | Media |
| PYTORCH | 1. Soporte nativo de Python 2. Optimización de memoria incorporada 3. Redes neuronales dinámicas 4. Aprovechamiento de Recursos (GPU) | Fácil |
| KERAS | 1. Modularidad 2. Producción de modelos para teléfonos inteligentes (iOS y Android) 3. Transparente sobre TensorFlow 4. Herramientas para facilitar el trabajo con datos de imagen y texto. 5. Código Libre 6. Trabaja con Python 7. Rendimiento al trabajar con GPU y CPU. | Fácil |

Tabla 6: Comparación entre los distintos Frameworks.

# **DESARROLLO**

En esta sección se describe el diseño de un Asistente Virtual Basado en Visión por Computadora, el cual es una aplicación que está basada en técnicas de reconocimiento facial y aprendizaje automático para la identificación de rostros humanos. El sistema esta conformado por distintos módulos que operan de manera sistemática sobre la información obtenida de video.

## ENTORNO DE TRABAJO

Para llevar a cabo el proyecto ha sido necesario realizar la preparación del entorno de trabajo en el cual hemos procedido a instalar diversas herramientas. El sistema operativo utilizado fue Windows 10 en el realizamos la instalación de OpenCV 2.4, así como Keras y TensorFlow que fueron mencionados en la sección anterior, además de la instalación del lenguaje de programación Python 3 y el editor de código Visual Studio Code desarrollado por Microsoft y disponible para Windows, Linux y macOS.

El software fue desarrollado en un computador con las siguientes características:

* Procesador: 3.50 GHz Intel Core i7 7500U
* Memoria RAM: 16 GB DDR4
* Tarjeta de Video: NVIDIA GeForce GTX 940MX 2GB
* Disco Duro: HDD 1 TB SATA 3Gb/s, 5400 RPM, 8 Mb Cache

## DEFINICIÓN DE REQUERIMIENTOS

Para llevar a cabo la realización del presente proyecto, se propone la siguiente metodología, Figura 15. Con la finalidad de cumplir cada uno de los objetivos. A continuación, se presenta el desglose por etapas de las actividades que conducirán el trabajo.

Figura 17: Descripción de la metodología.

## ANÁLISIS DE REQUERIMIENTOS

El análisis de requerimientos trata de establecer lo que el sistema debe de hacer, sus propiedades emergentes deseadas y esenciales además de las restricciones en el funcionamiento del sistema y los procesos de desarrollo del software. En otras palabras, la captura de requisitos es la actividad mediante la que el equipo de desarrollo de un sistema de software extrae, de cualquier fuente de información disponible, las necesidades que debe cubrir dicho sistema.

Requerimiento General

La finalidad del proyecto a realizar es el de implementar un software que permita, a través de algoritmos y librerías, poder reconocer el rostro de una persona por medio de video capturado en tiempo real, tomado desde una cámara web, para permitir una atención a invitados cuando el administrador se encuentre ausente, además de un gestor de notificaciones o agenda y reconocimiento de voz.

Requerimientos funcionales

* Implementación de algoritmo para la detección de rostro
* Implementación de algoritmo para el reconocimiento facial.
* Añadir y eliminar imágenes desde la base de datos
* Mostrar en una pantalla de Windows la cámara web para la prueba del software
* Crear una base de datos local en la cual se almacenará cada imagen y notificación

Requerimientos no funcionales

* Diseño de una interfaz de usuario sencilla.
* Rendimiento de software.
* La base de datos como se implementará de forma local siempre estará disponible.
* Integridad de las imágenes en la base de datos
* Fácil interacción con el usuario

## CODIFICACIÓN DE ALGORITMOS

PREPROCESADO

EXTRACCIÓN DE CARACTERÍSTICAS

REDUCCIÓN DE DIMENSIONALIDAD

CLASIFICADORES

## CONSTRUCCIÓN DEL ASISTENTE VIRTUAL

## PRUEBAS, DEPURACIÓN DE ALGORITMOS Y AJUSTES DEL ASISTENTE

# **RESULTADOS**

El asistente realiza adecuadamente el Reconocimiento e Identificación de un usuario propietario en tiempo real, marcando las características propias de cada usuario con un porcentaje de acierto calculado a partir de los usuarios registrados como se muestra en la figura 2.

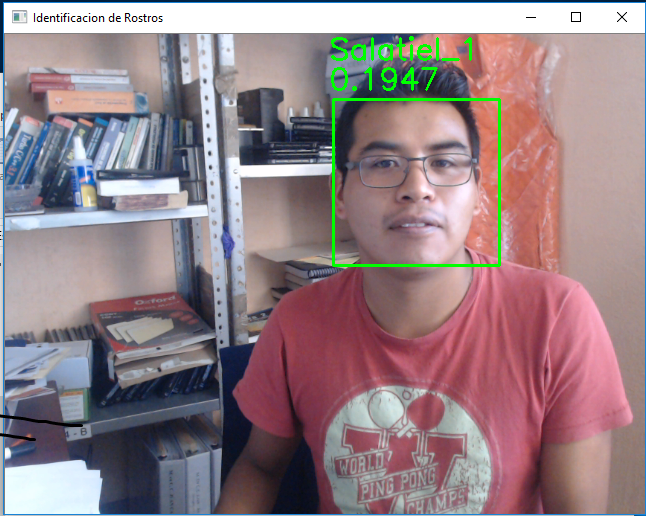


Figura 18. Identificación de rostro

* Actualmente se está trabajando en la implementación de un algoritmo para la conversión de audio a texto lo cual permita registrar los mensajes que se dejan para el usuario propietario.

# **CONCLUSIONES Y RECOMENDACIONES**

# **REFERENCIAS**

|  |  |
| --- | --- |
| [1] | MIT, «MIt Technology Review,» MIT Technology, 24 February 2009. [En línea]. Available: http://www2.technologyreview.com/news/412191/tr10-intelligent-software-assistant/. [Último acceso: 01 01 2019]. |
| [2] | C.-H. Chien, «CrunchBase,» [En línea]. Available: https://www.crunchbase.com/person/chi-hua-chien. [Último acceso: 02 01 2019]. |
| [3] | 5. e. o. s. recognition, «NEWS CHECK,» WELT, 20 04 2012. [En línea]. Available: https://www.welt.de/newsticker/dpa\_nt/infoline\_nt/computer\_nt/article106206488/Von-IBM-Shoebox-bis-Siri-50-Jahre-Spracherkennung.html. [Último acceso: 01 01 2019]. |
| [4] | W. C. A. C. H. M. i. t. S. Market, «TheStreet,» 18 03 2014. [En línea]. Available: https://www.thestreet.com/story/12534433/1/why-cortana-assistant-can-help-microsoft-in-the-smartphone-market.html. [Último acceso: 01 01 2019]. |
| [5] | Apple, «Use Siri on all your Apple devices,» 01 12 2019. [En línea]. Available: https://support.apple.com/en-us/HT204389. |
| [6] | A. Alexa, «Amazon Alexa,» 01 01 2019. [En línea]. Available: https://developer.amazon.com/alexa-skills-kit?&sc\_category=Owned&sc\_channel=WB&sc\_publisher=Website&sc\_content=Content&sc\_detail=CTA&sc\_funnel=Visit&sc\_country=US&sc\_medium=Owned\_WB\_Website\_Content\_CTA\_Visit\_US\_SiteVisitors&sc\_segment=SiteVisitors. |
| [7] | Cognex, «Introduccion a la vision artificial,» 3 01 2019. [En línea]. Available: http://www.ikusmen.com/documentos/descargas/3cbb38\_Introduction%20to%20Machine%20Vision.pdf. |
| [8] | G. A. L. Mantilla, Diseño de un sistema de vision artificial para la revision del nivel de llenado de bebidas embotelladas, Barranquilla: Universidad Autonoma del Caribe, 2015. |
| [9] | Neoteo, «Xatakafoto,» 18 12 2009. [En línea]. Available: https://www.xatakafoto.com/camaras/sensores-con-tecnologia-ccd-vs-cmos. |
| [10] | W. y. Xiaoguang, Research on Text Categorization Based on Machine Learning. |
| [11] | Mark y Eibe, Data Mining: Practical Machine Learning Tools and Techniques, p. 232. |
| [12] | R. Baeza-Yates y B. Ribeiro-Neto, Modern Information Retrieval, Nueva York: Addison-Wes­ley, 1999. |
| [13] | F. B. L. B. S. W.-T. R. Crick, General Nature of the Genetic Code for Proteins, 1961. |
| [14] | F. Rosenblatt, Principles of Neurodynamics, 1961. |
| [15] | Minsky & Pappert , Perceptrons, 1969. |
| [16] | J. Parraga-Alava, Implementation of Artificial Neural Network, Santiago de Chile, 2013. |
| [17] | F. J. P. Burgos, «Redes Neuronales con GNU/Linux,» 01 01 2019. [En línea]. Available: http://es.tldp.org/Presentaciones/200304curso-glisa/redes\_neuronales/curso-glisa-redes\_neuronales.pdf. |
| [18] | D. Calvo, «Clasificación de redes neuronales artificiales,» 13 07 2017. [En línea]. Available: http://www.diegocalvo.es/clasificacion-de-redes-neuronales-artificiales/. |
| [19] | J. U. Burgal, «Deep Learning básico con Keras,» 11 07 2018. [En línea]. Available: https://enmilocalfunciona.io/deep-learning-basico-con-keras-parte-2-convolutional-nets/. |
| [20] | Matlab, «Redes Neuronales Convolucionales,» 01 01 2019. [En línea]. Available: https://la.mathworks.com/solutions/deep-learning/convolutional-neural-network.html. |
| [21] | D. Calvo, «Red Neuronal Convolucional CNN,» 20 07 2017. [En línea]. Available: http://www.diegocalvo.es/red-neuronal-convolucional/. |
| [22] | S. Sharma, «Activation Functions,» 6 09 2017. [En línea]. Available: https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6. |
| [23] | Prabhu, «Understanding of Convolutional Neural Network,» 04 03 2018. [En línea]. Available: https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148. |
| [24] | Oquab, Learning and transferring mid-level image representations using convolutional neural networks, 2014. |
| [25] | Matlab, «Deep Learning,» 01 01 2019. [En línea]. Available: https://la.mathworks.com/discovery/deep-learning.html. |
| [26] | B. L. G. H. V.C. Gungor, Opportunities and challenges of Wireless Sensor Networks in Smart Grid, IEEE Transactions on Industrial Electronics, October 2010. |
| [27] | O. M.Parkhi, A. Vedaldi y A. Zisserman, Deep Face Recognition, University of Oxford. |
| [28] | H. a. W. H. El Khiyari, Face Recognition across Time Lapse Using Convolutional Neural Networks., Journal of Information Security, 2016. |
| [29] | Amazon, «Amazon,» [En línea]. Available: https://aws.amazon.com/es/nosql/. [Último acceso: 10 11 2018]. |
| [30] | Oracle, «Oracle Store,» [En línea]. Available: https://shop.oracle.com/apex/product?p1=OracleDatabase&p2=&p3=&p4=&p5=. [Último acceso: 12 10 2018]. |
| [31] | «MYSQL,» MYSQL Server, [En línea]. Available: https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html. [Último acceso: 17 10 2018]. |
| [32] | MongoDb, «MongoDB Documentation,» [En línea]. Available: https://docs.mongodb.com/manual/introduction/. [Último acceso: 2 09 2018]. |
| [33] | «Acerca de Python,» Fundación de Software Python, [En línea]. Available: https://www.python.org/about/. [Último acceso: 11 12 2018]. |
| [34] | S. Jackman, «R Para el Metodólogo Político,» de El metodólogo político, Heather L. Ondercin, Pennsylvania State University, 2003, p. 40. |
| [35] | Oracle, «The Java Language Environment,» Oracle, [En línea]. Available: https://www.oracle.com/technetwork/java/intro-141325.html. [Último acceso: 10 12 2018]. |
| [36] | «Nvidia Developer,» Nvidia, [En línea]. Available: https://developer.nvidia.com/deep-learning-frameworks. [Último acceso: 29 12 2018]. |
| [37] | «TensorFlow,» [En línea]. Available: https://www.tensorflow.org/. [Último acceso: 29 12 2018]. |
| [38] | Y. a. S. E. a. D. J. a. K. S. a. L. J. a. G. R. a. G. S. a. D. T. Jia, «Caffe: Convolutional Architecture for Fast Feature Embedding,» 2014. [En línea]. [Último acceso: 20 11 2018]. |
| [39] | B. Lorica, «Why AI and machine learning researchers are beginning to embrace PyTorch,» 11 12 2017. [En línea]. Available: https://www.oreilly.com/ideas/why-ai-and-machine-learning-researchers-are-beginning-to-embrace-pytorch. [Último acceso: 12 11 2018]. |
| [40] | Keras, «Keras.io,» [En línea]. Available: https://keras.io/#why-this-name-keras. [Último acceso: 30 12 2018]. |
| [41] | P. Flach, Machine Learning: The Art and Science of Algorithms that Make Sense of Data., Cambridge University Press, 2012. |

# **ANEXOS**

from keras.models import Model, Sequential

from keras.layers import Input, Convolution2D, ZeroPadding2D, MaxPooling2D, Flatten, Dense,Dropout, Activation

from PIL import Image

import numpy as np

from keras.preprocessing.image import load\_img, save\_img, img\_to\_array

from keras.applications.imagenet\_utils import preprocess\_input

from keras.preprocessing import image

import matplotlib.pyplot as plt

from keras.models import model\_from\_json

import tensorflow as tf

import keras

import cv2

from os import listdir

import cv2, imutils, os, time, shutil, time

#import Vistas.interfaz\_admin as admin

import Vistas.cliente as cliente

import datetime

import Vistas.logica\_cliente as cli

#------------------------------------------------------------------------------------------

#################Configuracion del CPU-GPU

config = tf.ConfigProto( device\_count = {'GPU': 1 , 'CPU': 56} )

sess = tf.Session(config=config)

keras.backend.set\_session(sess)

#------------------------------------------------------------------------------------------

##############################################################################################

#########################PARAMETROS GENERALES DEL PROGRAMA####################################

#Titulo de la ventana de OpenCV

title\_windows = 'Identificacion de Rostros'

#Cargamos clasificador de caras frontales

face\_cascade = cv2.CascadeClassifier('./CNN/Assets/haarcascade\_frontalface\_default.xml')

#Seleccionamos el tamaño de la imagen de entrada

size\_height, size\_width = 224, 224

#Cargamos la bases de datos de imagenes

#En esta ubicacion se alojan las imagenes que se quieren reconocer con extension jpg

database\_images = './CNN/database/Usuarios\_Registrados/'

#Color de la linea que es mostrada en OpenCV

color = (0,255,0)

#Ruta de pesos https://drive.google.com/file/d/1CPSeum3HpopfomUEK1gybeuIVoeJT\_Eo/view

weights = './CNN/Assets/vgg\_face\_weights.h5'

##Instancia de la camara

camara = cv2.VideoCapture(0)

################################################################################################

url = 'http://127.0.0.1:8081/'

################################################################################################

def preprocess\_image(image\_path):

#Procedemos a tratar las imagenes como vector

#Cargamos la imagen y establecemos un tamaño de entrada

img = load\_img(image\_path, target\_size=(size\_width, size\_height))

#Convertimos la imagen en array

img = img\_to\_array(img)

#Se agrega una nueva columna al vector de imagen

img = np.expand\_dims(img, axis=0)

#Adecuamos la imagen al formato que requiere el modelo

img = preprocess\_input(img)

return img

def loadVggFaceModel():

#Aplicamos transferencia de aprendizaje segun http://www.robots.ox.ac.uk/~vgg/publications/2015/Parkhi15/parkhi15.pdf tabla 3

model = Sequential()

#Tamaño de la imagen 224x244 x 3 que es el RGB

model.add(ZeroPadding2D((1,1),input\_shape=(size\_width, size\_height, 3)))

#Bloque 1

model.add(Convolution2D(64, (3, 3), activation = 'relu'))

model.add(ZeroPadding2D((1, 1)))

model.add(Convolution2D(64, (3,3), activation= 'relu'))

model.add(MaxPooling2D((2, 2), strides=(2,2)))

#Bloque 2

model.add(ZeroPadding2D((1,1)))

model.add(Convolution2D(128,(3,3),activation='relu'))

model.add(ZeroPadding2D((1,1)))

model.add(Convolution2D(128, (3,3), activation='relu'))

model.add(MaxPooling2D((2,2), strides=(2,2)))

#Bloque 3

model.add(ZeroPadding2D((1, 1)))

model.add(Convolution2D(256, (3,3),activation='relu'))

model.add(ZeroPadding2D((1,1)))

model.add(Convolution2D(256, (3,3), activation='relu'))

model.add(ZeroPadding2D((1,1)))

model.add(Convolution2D(256, (3,3), activation='relu'))

model.add(MaxPooling2D((2,2), strides=(2,2)))

#Bloque 4

model.add(ZeroPadding2D((1,1)))

model.add(Convolution2D(512, (3,3), activation='relu'))

model.add(ZeroPadding2D((1,1)))

model.add(Convolution2D(512, (3,3), activation='relu'))

model.add(ZeroPadding2D((1,1)))

model.add(Convolution2D(512, (3,3), activation='relu'))

model.add(MaxPooling2D((2,2), strides=(2,2)))

#Bloque 5

model.add(ZeroPadding2D((1,1)))

model.add(Convolution2D(512, (3,3), activation='relu'))

model.add(ZeroPadding2D((1,1)))

model.add(Convolution2D(512, (3,3), activation='relu'))

model.add(ZeroPadding2D((1,1)))

model.add(Convolution2D(512, (3,3), activation='relu'))

model.add(MaxPooling2D((2,2), strides=(2,2)))

#Bloque 6

model.add(Convolution2D(4096, (7,7), activation='relu'))

model.add(Dropout(0.5))

model.add(Convolution2D(4096, (1,1), activation='relu'))

model.add(Dropout(0.5))

model.add(Convolution2D(2622, (1,1)))

model.add(Flatten())

model.add(Activation('softmax'))

#Ingresamos los datos al modelo

vgg\_face\_descriptor = Model(inputs = model.layers[0].input, outputs = model.layers[-2].output)

#Cargamos pesos ya pre-entrenado

model.load\_weights(weights)

#vgg\_face\_descriptor.load\_weights(weights)

return vgg\_face\_descriptor

def findSimilarity(source\_image, representation):

#Utilizamos Distancia del coseno para detectar las similitudes entre vectores

#https://sefiks.com/2018/08/13/cosine-similarity-in-machine-learning/

a = np.matmul(np.transpose(source\_image), representation)

b = np.sum(np.multiply(source\_image, source\_image))

c = np.sum(np.multiply(representation, representation))

return 1 - (a / (np.sqrt(b)\*(np.sqrt(c))))

#Cargamos el modelo

model = loadVggFaceModel()

#Variable donde se van a cargar los datos de las imagenes

array\_images = dict()

#Procedemos a tratar los nombres de la imagenes

for file in listdir(database\_images):

#Dividimos el nombre de la imagen de su extension

images, extension = file.split(".")

path = database\_images+'%s.jpg' % (images)

array\_images[images]= model.predict(preprocess\_image(path))[0,:]

print("Servicio Iniciado.")

exit = True

name =''

user = 0

list\_access = []

while(exit):

#Empezamos la lectura de video

\_, img = camara.read()

#Detectamos solo caras frontales

faces = face\_cascade.detectMultiScale(img, 1.5, 5)

#Pintamos las caras detectadas

for (x,y,w,h) in faces:

#Dibijamos un rectangulo para mostrar la cara

cv2.rectangle(img,(x,y),(x+w, y+h), color, 2)

#Seleccionamos las posiciones de la cara encerrada en el rectangulo

detected\_face = img[int(y):int(y+h), int(x):int(x+w)]

##Guardamos el rostro detectado con nuevas medidas

detected\_face = cv2.resize(detected\_face, (size\_width, size\_height))

#Convertimos la imagen a matriz

img\_pixels = image.img\_to\_array(detected\_face).astype(np.float32)

#Unimos la matriz

img\_pixels = np.expand\_dims(img\_pixels, axis= 0)

# Hacemos una prediccion

captured\_reprentation = model.predict(img\_pixels)[0,:]

found = 0 #Variable para guardar los caras identificadas

acierto = 0.0

for i in array\_images:

#Renombramos a 'i' con la variable que contiene el nombre de la persona

image\_name = i.split('\_')[0]

representation = array\_images[i]

#Metodo encargado de encontrar las similitudes con el metodo de distancia del coseno

similarity = findSimilarity(representation, captured\_reprentation)

#Evaluamos el porcentaje de similitud

acierto=(round(similarity,2))

if(acierto < 0.27): # Si es menor que 0.27 es cara detectada y procedemos a dibijar el nombre y porcentaje de la similitud

acierto\_str = str(abs(round(similarity,2)-100))+ ' %'

print('Es {} en {} %'.format(image\_name,abs(round(similarity,2)-100)))

cv2.putText(img,acierto\_str,(int(x+w-190), int(y-10)), cv2.FONT\_HERSHEY\_SIMPLEX, 1,color, 2)

cv2.putText(img,image\_name,(int(x+w-140), int(y-40)), cv2.FONT\_HERSHEY\_SIMPLEX, 1, color, 2)

found = 1

user = 0

list\_access.append(abs(round(similarity,2)-100))

break

#Variables para obtener la fecha

date = datetime.datetime.now()

name = str(date).split('.')[1]

#Verificamos los rostros presentes en la imagen

if len(faces) > 2:

print('Se detectaron {} rostros.'.format(len(faces)))

database\_images = './CNN/database/Usuarios\_Desconocidos/muchas\_{}.jpg'.format(name)

user = 0

del list\_access[:]

cv2.imwrite(database\_images, img)

#Comparamos la lista para crear un promedio de la cara detectada y permitir el acceso

if found== 1:

if len(list\_access) > 5:

if (sum(list\_access) // len(list\_access)) > 95:

print('Desplegar panel de control')

#Si no es detectada algun rostro aumentamos contador y volvemos a relizar el proceso para comprobar que en verdad no es usuario registrado

# y pueda ser tratado como invitado

if found == 0:

user += 1

print("{}. Desconocido".format(user))

cv2.putText(img,'Invitado',((x+w-170), (y-10)), cv2.FONT\_HERSHEY\_SIMPLEX, 1, color, 2)

if user == 10:

database\_images = './CNN/database/Usuarios\_Desconocidos/{}.jpg'.format(name)

cv2.imwrite(database\_images, detected\_face)

user = 0

cli.startCliente(database\_images)

del list\_access[:]

#Titulo de la ventana

cv2.imshow(title\_windows, img)

#Si es presionado la tecla 'q' sales del programa

if cv2.waitKey(1) & 0xFF == ord('q'):

#if exit === 'q':

break

#Liberar Memoria

camara.release()

cv2.destroyAllWindows()