

TECNOLÓGICO NACIONAL DE MÉXICO

Instituto Tecnológico de Culiacán



Opción:

TITULACIÓN INTEGRAL TESIS PROFESIONAL

**“Clasificación de imágenes de galaxias mediante el uso de
redes neuronales convolucionales”**

ALUMNO:

René Cecil Vidríales Trujillo.

CARRERA:

Ingeniería en Sistemas Computacionales.

No. DE CONTROL:

15171223.

DIRECTOR:

Dr. Héctor Rodríguez Rangel.



DIVISION DE ESTUDIOS PROFESIONALES
ASUNTO: **Autorización de Impresión**

Culiacán, Sinaloa, 26/11/2020 12:46

C. RENÉ CECIL VIDRIALES TRUJILLO
PASANTE(S) DE LA CARRERA DE INGENIERÍA EN SISTEMAS COMPUTACIONALES
P R E S E N T E.

De acuerdo con fallo emitido por la Comisión Revisora Integrada por:

	Cédula Profesional
Presidente: Doctorado en Ciencias en Ingeniería Eléctrica Opción en Sistemas Computacionales Héctor Rodríguez Rangel	9972835
Secretario: Maestría en Ciencias en Ciencias de la Computación Gloria Ekaterine Peralta Peñuñuri	6038275
Vocal: Licenciado en Informática Nora Esmeralda Cancela García	3373654

Y considerando que reúne los requisitos establecidos en la Opción: **TITULACIÓN INTEGRAL: TESIS PROFESIONAL "CLASIFICACIÓN DE IMÁGENES DE GALAXIAS MEDIANTE EL USO DE REDES NEURONALES CONVOLUCIONALES"**, damos a usted nuestra autorización para que proceda a imprimir su trabajo profesional

Sin otro particular, quedo de usted.

ATENTAMENTE

Excelencia en Educación Tecnológica®

M.C. CARLOS RAFAEL LIZARRAGA ARREOLA
JEFE DE LA DIVISIÓN DE ESTUDIOS PROFESIONALES



EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA



TECNOLÓGICO
NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO DE CULIACÁN

DEPARTAMENTO DE DIVISIÓN DE
ESTUDIOS PROFESIONALES



Dedicatoria

A mi madre, Claudia. A mis hermanos, Laura y Luis. Y a mi papá, Manuel. Las mil y una noches de incertidumbre, percances económicos y carencias de comida no pudieron con nosotros. Siempre estaré agradecido por todo el apoyo y amor que me dieron en todo este camino. Me atrevo a decir que somos imparables y a prueba de todo, los amo por y para siempre.

A mis abuelos Cristina y José, a mis tíos Alma, Jorge, Felipe y Claudia. Gracias por estar siempre al pie del cañón, así como su apoyo y amor no solo para mí, si no también para con mi madre y hermanos. Verdaderamente sin ustedes este trayecto hubiera sido mucho más escabroso de lo que ya fue. A mis primos, que son como mis hermanos. Sin ustedes este recorrido hubiera sido gris y no tan divertido. Los amo a todos.

A mis amigos. Ustedes saben quiénes son. Les agradezco infinitamente todas las veces que fueron un soporte emocional, así como todas las experiencias que hemos tenido a lo largo de estos años.

Le extiendo un gran agradecimiento a mis docentes universitarios quienes me formaron académicamente hasta llegar a este punto, aportando conocimiento, consejos y valores para realizar mi trabajo de una forma profesional y moralmente correcta.

Le agradezco mucho a mi asesor y director de tesis, el Dr. Héctor Rodríguez Rangel y a mi codirectora de tesis, la M.C. Gloria Peralta Peñuñuri. Todavía recuerdo el día que pedí oportunidad para realizar mi servicio social con ustedes, estoy muy agradecido de haber podido llegar de la mano de ustedes hasta este momento. Sus consejos profesionales, académicos y de vida hicieron posibles la culminación de esto.

Y finalmente a Max, nuestro perro. Mi acompañante durante muchos desvelos.

CLASIFICACIÓN DE IMÁGENES DE GALAXIAS MEDIANTE EL USO DE REDES NEURONALES CONVOLUCIONALES

RESUMEN

Las galaxias constituyen uno de los objetos astronómicos más grandes en el universo observable y emblemáticos para la humanidad. Sus múltiples morfologías que dependen de diversos factores han sido durante muchos años el foco de la comunidad astronómica y científica, generando una gran cantidad de estudios, tesis y artículos. El estudio de estos cuerpos celestes es generalmente mediante simulaciones e imágenes obtenidas a través de telescopios, el problema de estas imágenes radica en la alta cantidad de ruido digital que contienen, haciendo compleja la tarea de clasificación de estas. En este trabajo se propone la implementación de un sistema basado en redes neuronales convolucionales, en el cual se extraerán solamente las características más importantes de las imágenes mediante autoencoders con el afán de eliminar el ruido digital, para posteriormente clasificarlas con una red neuronal convolucional. Este trabajo aporta una solución a los problemas existentes con la clasificación empírica. Se hace uso de herramientas de *Python* en conjunto con librerías proporcionadas por *Keras* y *Tensorflow*. Se utilizó un autoencoder para limpiar el ruido de las imágenes y una red neuronal convolucional (CNN) para clasificar las imágenes entre los distintos tipos de galaxias. Los experimentos se realizaron en un set de imágenes de galaxias. Los resultados experimentales dieron respuestas positivas en el set de datos de validación, logrando alcanzar un nivel bajo de error.

Palabras clave: Galaxias, ruido digital, red neuronal convolucional, autoencoder.

Contenido

1 INTRODUCCIÓN.....	7
1.1 Antecedentes	8
1.2 Descripción del problema	10
1.3 Hipotesis o preguntas de investigación	10
1.4 Objetivo general	11
1.5 Objetivos específicos	11
1.6 Justificación	11
1.7 Alcance	12
2 FUNDAMENTO TEÓRICO	13
2.1 Galaxia	13
2.2 Inteligencia artificial	14
2.3 Red neuronal convolucional	15
2.4 Aumento de datos	18
2.5 Ruido digital	20
2.6 Autoencoder.....	21
3 PROCEDIMIENTO	24
3.1 Descripción de las etapas	24
3.2 Funcionamiento general	25
3.3 Datos a utilizar	26
3.4 Aumento y uniformidad de datos	28
3.5 Diseño de autoencoder	34
3.6 Diseño de red neuronal convolucional	36
3.7 Experimento de entrenamiento de red neuronal convolucional con imágenes sin aumento de datos y sin limpieza de ruido digital.....	36
3.8 Experimento de entrenamiento de red neuronal convolucional con imágenes después de aumento de datos sin limpieza de ruido digital	41
3.9 Limpieza de imágenes.....	42
3.10 Entrenamiento de red neuronal convolucional con imágenes después de aumento de datos y después de limpieza de ruido digital	44
4 RESULTADOS	46
5 CONCLUSIONES.....	48
6 COMPETENCIAS DESARROLLADAS	50
7 FUENTES DE INFORMACIÓN	51

Lista de figuras

Figura 1 Galaxia elíptica ESO 325 (izq.) y galaxia espiral Messier 101.....	13
Figura 2 Matriz de entrada y detector de características	17
Figura 3 Proceso de convolución	18
Figura 4 Imagen con poco ruido digital vs con ruido digital añadido	20
Figura 5 Galaxia IC5332	21
Figura 6 Autoencoder básico	22
Figura 7 Representación general del proceso de un autoencoder	22
Figura 8 Funcionamiento general del sistema	26
Figura 9 Galaxia elíptica ESO 483-13	27
Figura 10 Galaxia espiral NGC4321.....	28
Figura 11 Galaxia lenticular NGC5018.....	28
Figura 12 Rotación de galaxia espiral NGC628.....	29
Figura 13 Recorte de la galaxia espiral NGC628.....	29
Figura 14 Volteo de galaxia espiral NGC628 sobre eje X	30
Figura 15 Volteo de galaxia espiral NGC628 sobre eje Y	30
Figura 16 Zoom de galaxia espiral NGC628.....	30
Figura 17 Aplicación de Dropout	38
Figura 18 Primer experimento, exactitud e inexactitud del modelo	41
Figura 19 Segundo experimento, exactitud e inexactitud del modelo	42
Figura 20 NGC3351, antes y después de su limpieza	43
Figura 21 Tercer experimento, exactitud e inexactitud del modelo.....	44

Lista de algoritmos

Algoritmo 1 Generador de transformaciones.....	31
Algoritmo 2 Función para transformar imágenes.....	32
Algoritmo 3 Generador de imágenes de Keras.....	33
Algoritmo 4 Autoencoder	34

Lista de tablas

Tabla 1 Primer experimento, porcentajes finales de exactitud e inexactitud.....	42
Tabla 2 Segundo experimento, porcentajes finales de exactitud e inexactitud	43
Tabla 3 Tercer experimento, porcentajes finales de exactitud e inexactitud.....	45

1 INTRODUCCIÓN

Actualmente la inteligencia artificial se encuentra de alguna forma u otra en casi cualquier actividad que se realiza por medio de la tecnología, su adopción a gran escala ha hecho posible el desarrollo de librerías o herramientas de trabajo con las cuales es sencillo aplicar técnicas de inteligencia artificial a problemas existentes. Esto da un paso enorme a que áreas distintas a las de las ciencias computacionales tomen en cuenta a la inteligencia artificial como una herramienta poderosa que les puede servir como apoyo para hacer de sus tareas cotidianas algo más llevadero y sencillo. En el caso de la astronomía, se tienen muchas áreas de oportunidad que pueden ser atacadas con inteligencia artificial, siendo una de ellas la clasificación de imágenes de galaxias.

Esta área de oportunidad da paso a la creación de clasificadores basados en redes neuronales capaces de recibir una imagen de entrada y dar una predicción sobre a qué tipo de galaxia pertenece esta imagen. De las características más importantes que conforman a este clasificador se encuentra un autoencoder, el cual es un tipo de red neuronal que se encarga de recibir una imagen, tomar sus características más importantes y reconstruirla, eliminando de manera efectiva las características no deseadas o menos importantes que se presenta, como por ejemplo el ruido digital. Es igualmente importante el proceso de aumento de información, en el cual se crean más ejemplares del conjunto de información en miras a tener información balanceada en cantidad, previniendo que la red neuronal convolucional aprenda de manera desigual o que ocurra el fenómeno de sobreajuste, que es cuando la red neuronal convolucional no logró generalizar bien las características del conjunto de información por lo tanto falla al momento de recibir entradas distintas a las que fue entrenada.

El análisis de imágenes mediante redes neuronales convolucionales permite obtener predicciones de manera automática, eliminando factores propios a los seres humanos que pueden intervenir de manera negativa en las tareas de clasificación. Debido a lo anterior se propone crear una red neuronal convolucional que identifique las distintas categorías de galaxia a las que una imagen puede pertenecer.

Este documento se divide de la siguiente manera: El fundamento teórico se encuentra en la sección 2. El procedimiento para la transformación y procesamiento de datos, diseño e implementación de redes neuronales convolucionales para autoencoders y clasificador se encuentra en la sección 3. En la

sección 4 se encuentran los resultados obtenidos durante el procedimiento. Finalmente, las conclusiones obtenidas durante el desarrollo del proyecto se plantean en la sección 5.

1.1 Antecedentes

Charles Messier identificó por primera vez una galaxia en el siglo 17, aunque en ese tiempo el desconocía qué eran. Messier era principalmente un observador de cometas, pero para evitar confundir a los cometas con las galaxias compiló una lista [14] con estos nuevos cuerpos. Esta lista contenía alrededor de 100 galaxias, pero pasaron casi 300 años para que los astrónomos averiguaran qué eran realmente los objetos de la lista de Messier. Unos años antes de que las galaxias acuñaran su nombre, Sir William Herschel y Sir John Herschel se dieron cuenta que estos cuerpos tenían diferentes formas y tamaños, y comenzaron a clasificar estos cuerpos basados en esas características.

Unos años más adelante, en el 1925, Edwin Hubble introdujo un diagrama [13] donde clasificaba a las galaxias de acuerdo con la forma de sus brazos y a su centro galáctico. En este diagrama Hubble sugería que las distintas formas de galaxias no eran más que parte de su proceso evolutivo, por lo tanto, este diagrama también relacionaba la edad de una galaxia con su forma. La nomenclatura introducida por Hubble en su diagrama sigue siendo utilizada hoy en día por los astrónomos, con la diferencia que hoy se sabe que la evolución de las galaxias no corresponde al diagrama, la evolución de estos objetos astronómicos es más escabrosa de lo que se pensaba.

Desde esos años hasta la actualidad la clasificación de galaxias se sigue haciendo de forma empírica y manual, gracias a los avances en tecnología ahora se tienen muchas más imágenes de galaxias que hace algunas décadas. Con esta gran cantidad de imágenes disponibles, se han hecho esfuerzos de carácter colaborativo donde cualquier persona puede recibir un pequeño entrenamiento en los distintos tipos de galaxias y posteriormente proceder a clasificar imágenes, un ejemplo de estos esfuerzos es Galaxy Zoo [15]. La cual es una herramienta colaborativa donde las personas que ingresan al sistema reciben un entrenamiento enfocado a la detección de galaxias y una vez concluido su entrenamiento pasan a la tarea real donde se les presentan imágenes de estos cuerpos astronómicos y realizan de forma individual pero colaborativa la clasificación de estos cuerpos astronómicos.

Otro tema que toma parte de esta investigación y también ha tenido múltiples tropiezos y aciertos es la inteligencia artificial. Esta rama de las ciencias de la computación tuvo unos inicios muy discordantes, opiniones encontradas y distintos puntos de vista sobre los cómo y porqué debería de emularse la inteligencia humana. La inteligencia artificial se compone de múltiples artefactos, teorías, herramientas y enfoques. Cada una de estas distintas herramientas tiene objetivos distintos que varían desde el procesamiento de lenguaje natural hasta la robótica. Una herramienta interesante de esta rama que toma un papel particularmente importante a esta investigación son las redes neuronales, las cuales de forma histórica han sido concebidas y empleadas como apoyo para problemas de clasificación.

Como primer punto a notar, las redes neuronales convolucionales para resolver problemas de clasificación de imágenes pueden describirse desde las arquitecturas de los primeros perceptrones utilizados para este fin, pero lo que las llevó a la fama que representan hoy fue una estructura de redes neuronales convolucionales llamada *LeNet*, propuesta en 1989 por Yann LeCun. En esta arquitectura LeCun se enfoca en resolver un problema de clasificación de códigos postales para el *USPS (United States Postal Service)* introduciendo términos como los mapas de características, con los cuales mejora la detección de características importantes de las imágenes independientemente de dónde están situadas en la imagen [23]. Con esta arquitectura LeCun abrió el camino para que las redes neuronales convolucionales fueran adaptadas para problemas complejos y distintos de visión por computadora, lo cual nos lleva al siguiente punto, los autoencoders.

Los *autoencoders* son un tipo de red neuronal artificial que se utilizan para aprender representaciones intermedias de su información de entrada. Para efectos de este proyecto, son especialmente útiles para hacer una reducción del ruido digital del conjunto de información que tenemos. La historia de los *autoencoders* no es muy clara, como son una variación de las redes neuronales, es posible decir que su historia data de los inicios de las redes neuronales. Generalmente a lo largo de la historia de este tipo de redes neuronales, se han utilizado para reducir las dimensiones de los conjuntos de información o detección de características importantes.

1.2 Descripción del problema

Las galaxias son uno de los objetos astronómicos más representativos para la humanidad, estos cuerpos celestes están conformados de gas, polvo y estrellas, todo esto agrupado en estructuras muy características gracias a la gravedad. Existen múltiples tipos de galaxias, variando la cantidad de estrellas que tiene la galaxia, sus espirales o ausencia de ellas, entre otras características variables.

A lo largo del tiempo, distintas agencias espaciales han liberado miles de fotografías de galaxias, aumentando la carga de trabajo de la comunidad científica debido a que dichas imágenes tienen que pasar por un procesamiento para reducir el ruido para finalmente poder clasificarlas de manera empírica. Dicho esto, existen herramientas propias de la inteligencia artificial que se han demostrado útiles con anterioridad en tareas de clasificación y procesamiento de imágenes en otras disciplinas.

Resulta necesario diseñar e implementar algoritmos de redes neuronales capaces de procesar y clasificar las imágenes de galaxias de forma eficiente y sencilla de tal manera que la carga de trabajo se reduzca para la comunidad científica y la certeza en la clasificación de las imágenes sea mayor.

1.3 Hipótesis o preguntas de investigación

La integración de las redes neuronales a áreas del conocimiento distintas a las ciencias de la computación ha servido fundamentalmente como apoyo a tareas realizadas de forma empírica o rudimentaria. Su versatilidad ha permitido que la inteligencia artificial esté disponible de maneras más sencillas y útiles al alcance de todos.

Una de las características más importantes de las redes neuronales convolucionales que sirven a este estudio son sus capacidades de predicción, así como la facilidad con la que se pueden manipular imágenes.

Este estudio, de manera principal, se enfoca en la clasificación de imágenes de galaxias mediante el uso de redes neuronales convolucionales. Las imágenes de galaxias se obtienen a través de las agencias espaciales internacionales, pero estas imágenes presentan un problema: tienen demasiado ruido digital y esto puede afectar gravemente el desempeño de la red neuronal convolucional. Con base a esto último, surge la siguiente pregunta: ¿Es posible generar una red neuronal convolucional confiable y

eficiente por medio de un conjunto de datos aumentado y filtrado por autoencoders capaz de clasificar apropiadamente a partir de una fotografía?

1.4 Objetivo general

Diseño e implementación de un clasificador de galaxias mediante el uso de técnicas de aprendizaje profundo como lo son los autoencoders y las redes neuronales convolucionales.

1.5 Objetivos específicos

- Diseñar e implementar una red neuronal convolucional capaz de clasificar una imagen entre los distintos tipos de galaxias.
- Diseñar e implementar una red neuronal capaz de extraer las características más importantes de las imágenes de tal forma que se pueda reconstruir una imagen con una menor cantidad de ruido digital.
- Diseñar e implementar una estrategia de aumento de información de tal forma que las distintas clases del set de datos estén balanceadas en cantidad.

1.6 Justificación

Hasta donde sabemos, no existe un análisis profundo sobre la combinación de herramientas basadas en redes neuronales para tareas de limpieza de ruido digital y clasificación en el campo específico de las galaxias. Previamente y en otras disciplinas ya se ha demostrado la efectividad de las redes neuronales para distintas tareas relacionadas con imágenes, en especial la clasificación asistida por computadora.

La evaluación de la información de galaxias publicada por distintas agencias espaciales, con un enfoque en la limpieza del ruido digital acompañada de la versatilidad y poder de las redes neuronales convolucionales proveerá a la comunidad científica un análisis imparcial sobre las imágenes de galaxias, otorgando un enfoque distinto a sobre como los especialistas clasifican estos objetos astronómicos de forma empírica.

En este proyecto se propone diseñar e implementar algoritmos de redes neuronales capaces de reducir el ruido digital en imágenes para posteriormente clasificar la fotografía de acuerdo con el tipo de galaxia. Esto es posible gracias a los patrones que presentan dichas imágenes.

1.7 Alcance

Se utilizará un conjunto de imágenes liberadas por las distintas agencias especiales, las cuales están conformadas por tres diferentes tipos de galaxias, conformadas de la siguiente manera:

- 39 imágenes de distintas galaxias lenticulares.
- 131 imágenes de galaxias espiral.
- 19 imágenes de galaxia elíptica.

Este conjunto de información se incrementará con técnicas de aumento de datos para poder conformar un set de datos equivalente entre sus distintas clases. Una vez el conjunto de datos sea equivalente, estas imágenes se procesarán con una red neuronal para quitarles el ruido digital y posteriormente se utilizarán para entrenar una red neuronal convolucional. Este modelo entrenado será embebido en un sistema que estará disponible en la web, de interfaz de usuario sencilla y amigable para que sea usado por el público interesado en clasificar imágenes de galaxias.

2 FUNDAMENTO TEÓRICO

El presente trabajo se centra en la clasificación de imágenes de galaxias mediante el uso de técnicas de inteligencia artificial. Para poder comprender este trabajo es necesario definir conceptos básicos que servirán de apoyo para interpretar mejor su lectura.

2.1 Galaxia

Las galaxias son uno de los cuerpos astronómicos más majestuosos y estudiados por la humanidad. Estos cuerpos fueron descubiertos hace varios cientos de años y son objeto de estudio desde entonces. Por lo anterior es correcto decir que se conoce mucho de ellas y existe una amplia gama de conceptos y definiciones derivadas de su estudio. Esto hace imperativo el exponer aquellas nociones y explicaciones que acerquen más a un concepto en general.

La vía láctea, como otras galaxias, está compuesta de una mezcla compleja de estrellas, planetas, gas interestelar, polvo, radiación y materia oscura [1]. Existen millones de galaxias en el universo observable, las cuales se clasifican dependiendo de su forma, edad y otros factores.

Usualmente las galaxias se clasifican en dos tipos básicos: elípticas y espirales. Larson (2002) [2] define a las galaxias elípticas como “de forma relativamente redonda y apoyadas principalmente por movimientos aleatorios” mientras que a las galaxias espirales las define como “discos planos apoyados principalmente por movimientos de rotación”. Lo cual nos sugiere que estas galaxias tienen orígenes distintos debido a los movimientos que mantienen a sus componentes unidos. A continuación, la Figura 1 pone en contraste una galaxia espiral con una galaxia elíptica.



Figura 1 Galaxia elíptica ESO 325 (izq.) y galaxia espiral Messier 101.

Tomando como base esta figura, podemos reparar en las claras diferencias entre los dos tipos principales de galaxias. La única similitud aparente que podemos encontrar son las estrellas y sus centros galácticos, lo que sí es muy variante es su forma. Las galaxias elípticas carecen de una forma definida, son irregulares en cuanto a forma y cantidad de estrellas, mientras las galaxias espirales tienen una forma bien definida y aparentemente poseen una gran cantidad de estrellas.

Las imágenes de la Figura 1 son posibles gracias a telescopios especializados en tomar imágenes del espacio profundo, como las tomadas por el telescopio espacial *Hubble* [3]. Este telescopio, es capaz de tomar múltiples imágenes del espacio exterior y con técnicas de posteriores de edición de imágenes componer imágenes aún más grandes donde se pueden observar hasta 265,000 [4] galaxias. A partir de estas imágenes del espacio profundo es de donde la comunidad científica obtiene imágenes como las vistas en la figura 1, las cuales sirven para la investigación de estos cuerpos celestes.

2.2 Inteligencia artificial

Visto lo anterior, es lógico decir que las galaxias son un buen objeto de estudio, y una de las formas en que podemos complementar su estudio es mediante técnicas de inteligencia artificial. El doctor Herbert A. Simon, define a la inteligencia artificial como “una rama de las ciencias de la computación que estudia las propiedades de la inteligencia mediante la sintetización de la inteligencia” [5].

Esta rama de las ciencias de la computación, aunque suene futurista, tiene bastantes años de existencia. En el año de 1956, John McCarthy y Marvin Minsky presentaron el primer programa de inteligencia artificial en el *Dartmouth Summer Research Project on Artificial Intelligence* [6], en esta conferencia se esperaba llegar a un consenso en diversos tópicos de la inteligencia artificial pero desafortunadamente no se pudo llegar a un acuerdo en cuanto a métodos o estándares de esta área. Lo que aconteció después de este suceso se le conoce como el invierno de la inteligencia artificial, el cual comprendió un periodo de varios años en los cuales tanto el interés como las inversiones gubernamentales enfocadas a la inteligencia artificial disminuyeron de una forma notable.

Uno de los primeros intentos para emular la inteligencia humana fueron los sistemas expertos, los cuales son unos artefactos diseñados para emular el razonamiento y las habilidades de toma de decisiones que tenemos los seres humanos, solo que los sistemas expertos suelen enfocarse a un área específica.

Independientemente del hecho anterior, la inteligencia artificial continuó progresando, se dice que a la par con los descubrimientos y desarrollo en hardware y software. En el lapso de los años 1957 a 1974, Joseph Weizenbaum creó y presentó el primer bot conversacional: ELIZA [7], de esta forma Joseph mostró el camino hacia la interpretación del lenguaje natural.

A los avances del día de hoy, se tienen dos definiciones vertientes de la inteligencia artificial, siendo inteligencia artificial general e inteligencia artificial débil. La primera define a una inteligencia artificial capaz de entender, aprender o llevar a cabo cualquier tarea mental o intelectual de la misma forma que los humanos lo hacemos. La segunda definición en cambio define a una inteligencia artificial que es experta o específica a un dominio, en el cual puede ocurrir que esta inteligencia supere a los humanos. A lo que concierne este estudio, se puede decir que el enfoque es basado en una inteligencia artificial débil, es decir, enfocado a un dominio en específico.

Con lo anterior entendemos que la inteligencia artificial es una rama de las ciencias de la computación que trata de emular de manera computacional a la inteligencia humana. El ejemplo anterior no es lo único que se ha logrado o que se puede hacer con inteligencia artificial, esta rama ha evolucionado enormemente, introduciéndose a otros ámbitos de la ciencia y siendo utilizada para fines distintos al procesamiento de lenguaje natural.

2.3 Red neuronal convolucional

Antes de introducir lo que son las redes neuronales convolucionales es necesario introducir otros términos y acepciones para poder comprender mejor lo que son estas herramientas. Primero que nada, las redes neuronales son un subconjunto del aprendizaje automático o aprendizaje máquina. El aprendizaje automático a su vez es un subconjunto de la inteligencia artificial que se enfoca en aprender generalidades y patrones de la información que se le presenta en vez de seguir reglas estructuradas de programación.

Ahora, un subconjunto del aprendizaje automático que es del cual salen los artefactos utilizados para la realización de esta investigación es el aprendizaje profundo o *deep learning*. El aprendizaje profundo es un subconjunto del aprendizaje automático que parte de las redes neuronales para su funcionamiento, el aprendizaje profundo tiene múltiples casos de uso, siendo los principales: procesamiento de imágenes y videos, reconocimiento de voz, diagnósticos de salud, videojuegos con historia no lineal. Las redes neuronales a las que se hace alusión se conocen como redes neuronales artificiales, se les conoce así para evitar su confusión con las redes neuronales biológicas, a las cuales se trata de emular.

Las redes neuronales artificiales imitan las conexiones existentes del cerebro y tienen un constructo llamado neurona, las cuales componen a la red y son encargadas de calcular valores a partir de la información de entrada y repartir esa información a través de sus conexiones. Estas redes se configuran para aplicaciones en específico, como reconocimiento de patrones o clasificación de datos.

Uno de los problemas principales de las redes neuronales artificiales es que debido a que su arquitectura es completamente distinta a la de los modelos de computación actuales, se tienen que emular, requiriendo cantidades grandes de poder de procesamiento para su entrenamiento, el hecho de querer, por ejemplo, utilizar un conjunto de información como imágenes para entrenar una red neuronal artificial se vuelve una tarea tediosa y de alto costo puesto que estas entidades solamente aceptan información en vectores simples, incrementando el número de parámetros que se deben de entrenar. Para problemas de clasificación de estructuras complejas, como las imágenes, es imperativo el uso de otro constructo, el cual en este caso vienen a ser las redes neuronales convolucionales.

Las redes neuronales convolucionales son un tipo de modelos de aprendizaje profundo, los cuales procesan información que está organizada en patrones cuadrículados, como las imágenes [8], para lo cual es necesario recordar que una imagen no es más que una matriz de píxeles. El objetivo de estos modelos es aprender características importantes de estas cuadrículas mediante convoluciones, las cuales, en términos simples, generan una serie de promedios.

Estas redes consisten en capas de entrada, de salida e intermedias. Las capas intermedias u ocultas generalmente se componen de capas de convolución, de agrupación (*pooling layers*) y finalmente capas completamente conectadas. Existen muchas variaciones de las arquitecturas, pero todas están basadas en el patrón mencionado arriba. Las distintas capas se describen de la siguiente forma:

1. La capa de entrada es el punto de inicio a la red neuronal convolucional. Para las redes neuronales convolucionales, la entrada generalmente es una matriz de pixeles, los cuales corresponden a una imagen. En esta matriz de pixeles se contiene también una profundidad, que corresponde a los canales de color.
2. Las capas ocultas están conformadas, a su vez, de otras capas, las cuales están encargadas de hacer la extracción de características de la matriz de pixeles, es decir, la imagen. Estas capas generalmente tienen un patrón repetitivo de convolución y agrupación, en los cuales de forma progresiva se van construyendo características de orden superior tomando como base el resultado de las convoluciones. Las convoluciones constan de tres elementos: la matriz de entrada (imagen), el detector de características y el mapa de características. En la Figura 2 se aprecia un ejemplo de una matriz de entrada y de un detector de características.

1	1	0	1	1	1	0	0	1
0	0	0	1	0	1	1	1	0
1	0	1	0	1	0	1	0	1
0	0	1	1	1	0	1	1	0
1	0	1	0	0	0	1	1	0
0	1	1	0	1	1	0	1	1
1	0	1	0	1	0	0	0	1
1	1	0	1	0	0	1	0	0
0	1	1	1	1	0	0	1	1

Matriz (imagen) de entrada

1	1	1
1	0	1
0	0	0

Detector de características

Figura 2 Matriz de entrada y detector de características

Al detector de carcterísticas también suele conocerse como *kernel* o filtro. Ahora, el proceso de convolución es muy sencillo:

- 2.1. Se posiciona el *kernel* sobre la matriz de entrada, en el lado superior izquierdo y se cuenta el número de celdas en el que ambas matrices son iguales.
- 2.2. El número de celdas iguales o coincidencias se agrega al mapa o matriz de

características, en la misma posición en la que se situó el *kernel* sobre la matriz de entrada.

2.3. Este proceso se repite hasta terminar con esa fila. Una vez que se termina la fila, se pasa a la siguiente, igual de izquierda a derecha, hasta completar el mapa de características. En la Figura 3 se muestra el resultado de un proceso de convolución en un mapa de características.

1	0	1	0
1	1	0	1
1	1	1	1
0	0	1	0

X

1	0
1	1

=

3	2
1	2

Matriz de entrada Kernel Mapa de características

Figura 3 Proceso de convolución

3. Finalmente, la capa o capas de salida. Estas capas pueden tomar distintos nombres, como capa de clasificación o capa completamente conectada. En esta o estas capas es donde se toma el mapa de características y se producen probabilidades o puntuaciones de clases basándose en el mapa. Aquí es donde la “predicción” o “clasificación” ocurren. Estas capas toman este nombre debido a que están completamente conectadas a las capas o neuronas anteriores.

Es preciso recalcar que el proceso de convolución se hace para extraer características importantes de las imágenes. Estas características que se pretende extraer dependen del *kernel* o detector de características, por lo tanto, existen distintos *kernel* o filtros con finalidades específicas, por ejemplo: resaltar bordes en objetos, remarcar zonas oscuras, suavizar el ruido digital, etc. Con las definiciones anteriores podemos llegar a una común definición sobre las redes neuronales convolucionales: son una clase de redes neuronales de aprendizaje profundo, utilizadas principalmente para el análisis de imágenes.

2.4 Aumento de datos

La inteligencia artificial trata de emular la inteligencia humana de forma computacional. Esta disciplina dispone y se compone de varias técnicas, como las redes neuronales. Una red neuronal aprende tomando como base la información con la que se alimenta, todo bien hasta aquí. El problema nace cuando se pretende entrenar a una red neuronal con poca información o información que no tiene mucha

diversidad. Cuando esto ocurre, ocasiona un problema llamado sobreajuste, el cual no es más que cuando el modelo de red neuronal funciona y aprende bien el conjunto de información de entrenamiento, pero falla al momento de trabajar sobre el conjunto de datos de prueba.

Según Ying [11], este problema ocurre por diversas razones, siendo las más representativas las siguientes:

1. Existe mucho ruido a lo largo del conjunto de datos de entrenamiento
2. El conjunto de datos de entrenamiento es muy pequeño
3. Existe una variación grande entre los datos representativos y el ruido a lo largo del conjunto de datos de entrenamiento.

Cuando el sobreajuste ocurre por la segunda razón, la raíz del problema es que debido a que se entrenó el modelo con poca información, el modelo no logra generalizar las características importantes o los patrones que tiene el conjunto de información, por lo tanto, al momento de probar el modelo con información de prueba su desempeño es pobre.

Este problema es muy conocido y se ha atacado a través de los años de diferentes formas: con arquitecturas de redes neuronales especiales para evitar este problema o con técnicas de aumento de datos. En este punto nos enfocaremos a las técnicas de aumento de datos.

Según May [12], “el aumento de datos es un método que genera más datos de entrenamiento tomando como base el conjunto de datos de entrenamiento existente.”. Generalmente las técnicas de aumento de datos se basan en operaciones simples que se les aplican a las imágenes, por ejemplo: cortar, rotar, voltear, mover sobre un eje, etc. Las técnicas de aumento de datos, bien aplicadas, nos apoyan en el proceso de conformar un conjunto de información equivalente cuando nuestro conjunto de datos actual es reducido o no es equivalente entre sus clases.

Las técnicas de aumento de datos se suelen utilizar mucho en dominios donde no es muy sencillo obtener grandes cantidades de información, como en el caso de ciertos estudios médicos o en el dominio

de la astronomía, donde si bien existen muchas imágenes de diversos cuerpos astronómicos, no todas pueden ser utilizadas para entrenar una red neuronal convolucional debido a su baja calidad.

2.5 Ruido digital

El ruido digital es un fenómeno que degrada la calidad de los archivos multimedia digitales. Entre los que tenemos imágenes, audio, videos, etc. En este caso, el exceso de ruido en las imágenes de las galaxias es preocupante y es un punto por atacar para poder obtener un clasificador óptimo.

El ruido varía de cámara a cámara y ocurre dependiendo de ciertos factores: sensibilidad, tiempo de exposición, temperatura del color, entre otros. Para conocer la magnitud del ruido existe un marcador llamado SNR (*Signal to Noise Ratio*) que indica la cantidad de ruido que existe comparado con la señal que se recibe, en el caso de las imágenes, la señal que se recibe es la luz que llega al sensor de la cámara. Entre más señal exista, menos ruido hay y viceversa. La Figura 4 muestra la comparación de una imagen con un ruido digital mínimo contra la misma imagen con exceso de ruido digital añadido.



Figura 4 Imagen con poco ruido digital vs con ruido digital añadido

Como podemos ver en la Figura 4 el ruido afecta gravemente la calidad de las imágenes. Enfocándonos en las imágenes de las galaxias, podemos decir que las imágenes tomadas a cuerpos celestes tienden a tener una alta cantidad de ruido digital, introducido principalmente por el alto tiempo de exposición al que se pone el lente. Esta alta exposición se debe a que es necesario capturar la mayor cantidad de señal (luz) para lograr obtener más detalle de los objetos fotografiados. De igual manera, se les introduce ruido en el post procesamiento porque hay situaciones en donde el ruido es benéfico para obtener características ocultas de las imágenes. La Figura 5 muestra la galaxia IC5332 donde se puede

notar a simple vista un alta cantidad de ruido, introducido principalmente por el post procesamiento y la alta exposición.



Figura 5 Galaxia IC5332

Lo interesante de este fenómeno son sus soluciones, la mayoría de ellas apoyadas en las matemáticas, otras apoyadas en técnicas innovadoras de fotografía como las fotos HDR o, como se prefirió solucionar: utilizando una especialización de las redes neuronales convolucionales, llamado autoencoder.

2.6 Autoencoder

Un autoencoder se puede definir como una técnica de aprendizaje no supervisado, que se encarga de aprender las representaciones más eficientes de la información. Según Baldi [9], “los *autoencoders* son circuitos simples de aprendizaje que buscan transformar la información de entrada a una salida con la menor distorsión posible.”

Los autoencoders se utilizan de manera general para extraer las características más importantes de la información de entrada, mediante un proceso llamado reducción de dimensiones, en el cual, Maaten y Postma [10] lo definen como “la transformación de una alta dimensión de información a una representación significativa, pero con su dimensión reducida, idealmente la representación reducida debe tener una dimensionalidad que corresponda intrínsecamente a la dimensión original de la información.” En este proceso, lo que se trata de hacer es que el autoencoder destruya y reconstruya la información de entrada, esperando que la reconstrucción se parezca lo más posible a la información original, esto debido a que el modelo del autoencoder está restringido a priorizar qué aspectos de la información deben persistir, aprendiendo solamente la información útil de los datos de entrada.

La estructura de estas entidades varía de tipo a tipo, pero la idea general es sencilla: un *encoder* toma el input, lo procesa y toma las características descritas por la implementación, acto seguido, un *decoder* toma estas características y las reconstruye tomando como base el dato original.

En la Figura 6 se puede apreciar la estructura de un autoencoder básico. La información de entrada se distribuye y procesa en una capa de neuronas, las cuales generalmente son una cantidad menor que las neuronas que componen la información de entrada, para finalmente obtener una salida compuesta de la misma forma que la información de entrada.

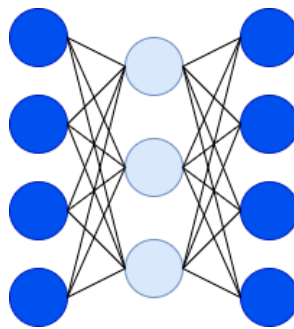


Figura 6 Autoencoder básico

En el lado derecho de la Figura 6 tenemos los nodos que conforman la información de entrada, en el centro se tienen los nodos de procesamiento interno, que generalmente son un número menor de nodos que los de entrada y finalmente se tiene una salida conformada por otro set de nodos que generalmente son iguales en cantidad que los nodos de entrada. En los nodos del centro es donde se lleva a cabo esta reducción de dimensionalidad, y en los nodos de salida se toma esta dimensionalidad reducida y se trata de crear una representación igual a la información de entrada. En la figura 7 se tiene una representación más general del funcionamiento del proceso por el que atraviesa una imagen al pasar por un autoencoder.

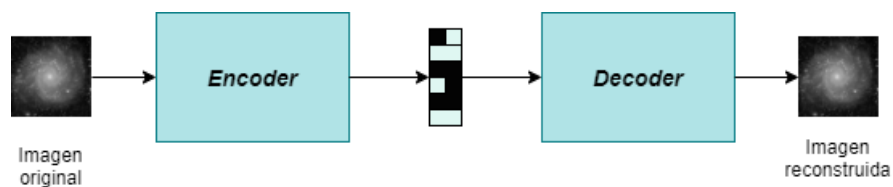


Figura 7 Representación general del proceso de un autoencoder

Estas entidades suelen ser útiles como apoyo en la tarea de la limpieza de la información previa al desarrollo y entrenamiento de las redes neuronales convolucionales, al limpiar el input que se entrega a las redes neuronales se obtiene un modelo entrenado más eficiente puesto que la información ya tiene un énfasis en las características más importantes.

3 PROCEDIMIENTO

En esta sección se describen las etapas ejecutadas para el desarrollo de este trabajo de investigación. En primera instancia se revisan de manera general los pasos efectuados, siguiéndoles una descripción a detalle de cada etapa para la realización de las redes neuronales que conformarán este clasificador de galaxias.

3.1 Descripción de las etapas

En primera instancia este trabajo de investigación consiste en una etapa de investigación, seguido de experimentación y desarrollo, para posteriormente implementar las pruebas necesarias y evaluar los resultados. En caso de que el proyecto sea exitoso de acuerdo con la hipótesis presentada, se procede al reporte de los resultados, para finalmente tener las conclusiones de este trabajo de investigación. De manera detallada, las etapas son descritas a continuación.

3.1.1 Etapa 1. Investigación.

- Revisión del estado del arte: Se realizó una investigación sobre el estado del arte para obtener las referencias y conocimientos necesarios sobre las galaxias, ruido digital, inteligencia artificial, sus técnicas y sobre el aumento de datos.
- Estudio del marco teórico: Se estudiaron los conceptos referentes al tema y se definió cuáles son aquellos que son necesarios para realizar el trabajo de investigación.

3.1.2 Etapa 2. Experimentación y desarrollo.

- Diseño del sistema: Se realizó un esquema del funcionamiento general de cómo fue el desarrollo del proyecto, desde la obtención de datos hasta la implementación de las redes neuronales para clasificar galaxias.
- Desarrollo del sistema: Se diseñaron e implementaron los algoritmos necesarios para filtrar, manipular y aumentar los datos, también se implementaron los algoritmos de redes neuronales convolucionales para la limpieza de los datos, así como para su predicción.

- Pruebas y validación de los resultados obtenidos: Se valoraron la precisión y pérdidas de las redes neuronales en miras de ajustar sus parámetros y mejorar la precisión.

3.1.3 Etapa 3. Conclusión y resultados.

- Evaluación de los resultados obtenidos de la experimentación previa: Se realizaron las pruebas necesarias donde se redujo el ruido digital de las imágenes, así como pruebas de aumento de información, con la finalidad de probar que los resultados obtenidos sean correctos para su implementación
- Análisis de la realización e implementación del proyecto: Se realizará un análisis del trabajo de investigación y se presentará una conclusión sobre las ventajas que ofrece la implementación de redes neuronales para la limpieza de los datos para la clasificación de imágenes de galaxias mediante redes neuronales convolucionales.

3.2 Funcionamiento general

El funcionamiento general del sistema es un proceso sencillo, el primer paso es tener las imágenes de galaxias, después estas imágenes pasarán por un proceso de aumento de datos en el cual las clases quedarán equivalentes en número de elementos. Después el aumento de datos, las imágenes pasan por un proceso en donde se reduce su ruido digital para después ajustar sus dimensiones y canales de color de tal manera que todo el conjunto de información tenga las mismas características.

Finalmente, con el conjunto de información aumentado y limpio se procede a entrenar la red neuronal convolucional y a guardar el modelo generado de tal forma que se pueda utilizar posteriormente para embeber este modelo en, por ejemplo, un sistema web al que se pueda acceder de manera sencilla para utilizar las capacidades de clasificación.

En el proceso de aumento de datos se aplican ciertos procedimientos que afectan a cada imagen de tal manera que la red neuronal convolucional aprenda características diferentes de la misma imagen, estos procedimientos aplican ciertas transformaciones como redimensión, recortes, rotaciones, etc. Al finalizar este procedimiento nuestro conjunto de información ve su número de elementos aumentado en cantidad.

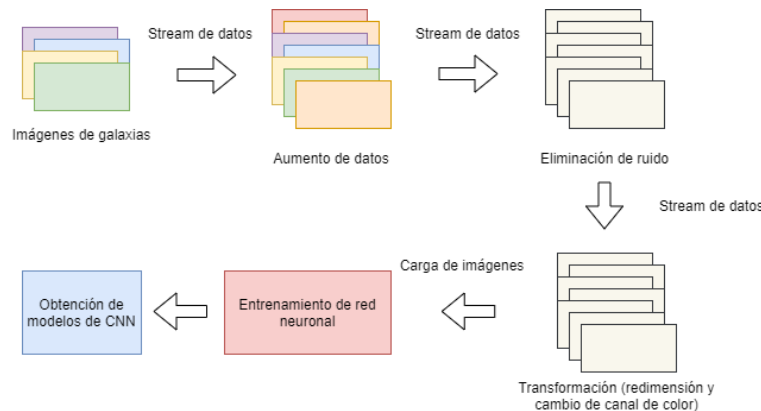


Figura 8 Funcionamiento general del sistema

Uno de los procedimientos más importantes de esta investigación es la limpieza de las imágenes, estas al contar con una cantidad grande de ruido digital puede pasar que la red aprenda características no importantes del conjunto de información, por lo tanto este proceso es necesario. Las imágenes al pasar por el *autoencoder*, se descomponen de tal forma que al reconstruirse nos quedemos solamente con las características más importantes de la imagen, reduciendo efectivamente el ruido digital.

Finalmente, las imágenes pasan por un procedimiento donde sus dimensiones y canal de colores de homologan de tal forma que este conjunto de información pueda utilizarse para el entrenamiento de la red neuronal convolucional. Una vez que el entrenamiento de la red concluye, su aprendizaje se extrae en unos modelos los cuales se pretende que sean utilizados por un sistema web para su uso en tiempo real y con una interacción sencilla de tal forma que los interesados en clasificar imágenes de galaxias tengan a su disposición un sistema confiable y eficiente para su uso.

3.3 Datos a utilizar

Las imágenes de galaxias se pueden obtener desde con telescopios comprados en cualquier tienda comercial hasta telescopios especializados de las distintas agencias espaciales que existen. Las imágenes de mejor calidad y más abundantes son las tomadas con telescopios especializados. Existen

telescopios como el *Hubble*, que se encuentran en la órbita de la tierra (esto se hace para evitar capturar la menor cantidad de ruido proveniente de la tierra) y existen otros telescopios como el *Very Large Telescope* que se encuentran en lugares distantes de la tierra, precisamente por las mismas razones que un telescopio se encuentra en la órbita de la tierra.

Basándonos en los diagramas de los distintos tipos de galaxias o en las clasificaciones existentes, nos dimos cuenta de que unas galaxias abundan más que otras, lo cual puede representar un problema debido a que la conformación de un set de información con una cantidad de datos equivalente entre clases es una prioridad dictada por el mismo proceso de aprendizaje que tienen las redes neuronales convolucionales.

Para los problemas de clasificación es necesario que cada categoría en el set de datos tenga una cantidad igual o parecida de información, de tal manera que la red neuronal aprende de manera equitativa de cada clase, evitando que los resultados sesguen a clases con una mayor cantidad de elementos en el entrenamiento. De igual manera se tuvo que poner en balanza la calidad de las imágenes existentes, existen galaxias con más cantidad en el universo, pero eso no necesariamente indica que sus imágenes sean de buena calidad. Debido a lo anterior, se obtuvo la siguiente cantidad de datos para los tipos de galaxias a continuación, los cuales serán el objetivo de este proyecto.

1. Galaxias elípticas (18 ejemplos)

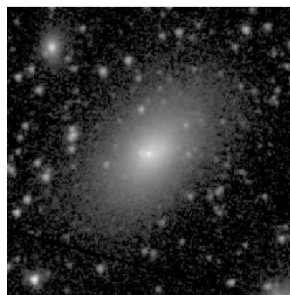


Figura 9 Galaxia elíptica ESO 483-13

2. Galaxias espirales (130 ejemplos)



Figura 10 Galaxia espiral NGC4321

3. Galaxias lenticulares (39 ejemplos)

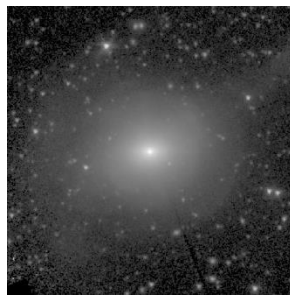


Figura 11 Galaxia lenticular NGC5018

Es de notar que no fue posible hacer sets de datos equivalentes para cada clase. Lo ideal hubiera sido hacer todas las clases equivalentes a una cantidad media, pero esto supondría que la clase más pequeña se vería reducida en información. Otra alternativa sería usar como máximo el número de datos totales que tiene la clase con menos información, pero en vista de que son 18 ejemplos, se prefirió dejar cada clase con su set de información íntegro y hacer optimizaciones al conjunto de información con técnicas de aumento de datos.

3.4 Aumento y uniformidad de datos

Basándonos en el punto anterior, contamos con un conjunto de información un poco reducido. Las redes neuronales convolucionales trabajan con cualquier cantidad de datos que se les dé, pero entre más datos se les entregue es mejor puesto que la diversidad en la información y los patrones que se encuentran en cantidades grandes de información hace que la red neuronal tenga un conocimiento generalizado sobre la información que se le está entregando. Cuando se les entrega poca información ocurre el fenómeno del sobreajuste debido a que cuando hay poca información la red generaliza patrones y características solamente dentro de ese pequeño conjunto de datos y como el conjunto de datos de prueba es distinto, la

red neuronal demuestra un desempeño pobre debido a que no aprendió de manera adecuada por la ausencia de información.

Existen distintas formas de atacar el sobreajuste, pero la solución más interesante que se decidió aplicar fue el aumento de datos. El aumento de datos es utilizado cuando no se tiene acceso a un conjunto de datos grande. Lo que hace el aumento de datos es aplicar distintas modificaciones a las imágenes de tal manera que la red neuronal reciba un input diferente o comprenda características diferentes de una misma imagen. A efectos de este proyecto se decidieron aplicar las siguientes técnicas de aumento de datos con el fin de tener un conjunto de datos equivalente en las tres clases utilizadas:

1. Rotación de imagen 10 grados en ambos ejes.

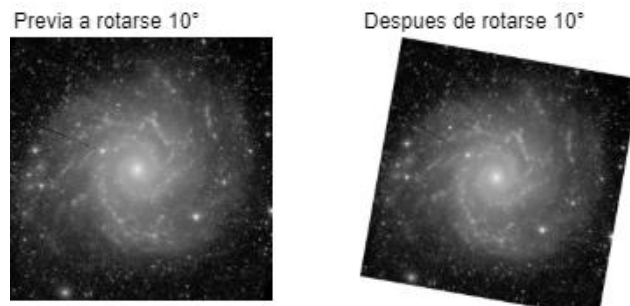


Figura 12 Rotación de galaxia espiral NGC628

2. Recorte de la imagen a un 10%.

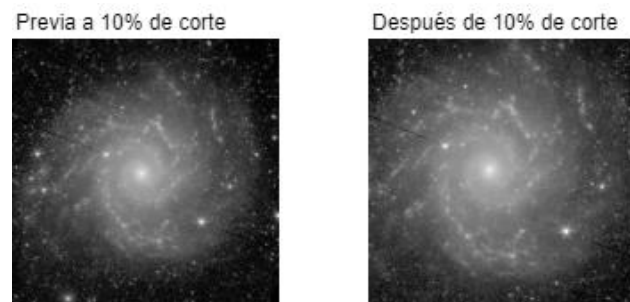


Figura 13 Recorte de la galaxia espiral NGC628

3. Voltear la imagen sobre el eje X.

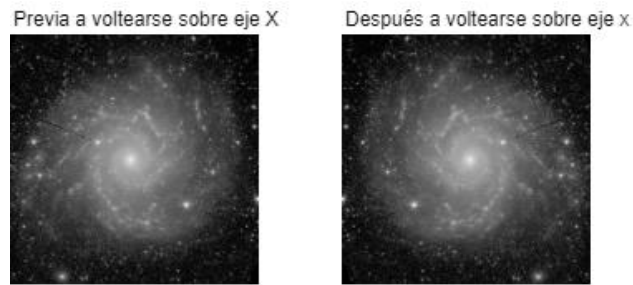


Figura 14 Volteo de galaxia espiral NGC628 sobre eje X

4. Voltear la imagen sobre el eje Y.



Figura 15 Volteo de galaxia espiral NGC628 sobre eje Y

5. Zoom a la imagen de 10% en cada eje.

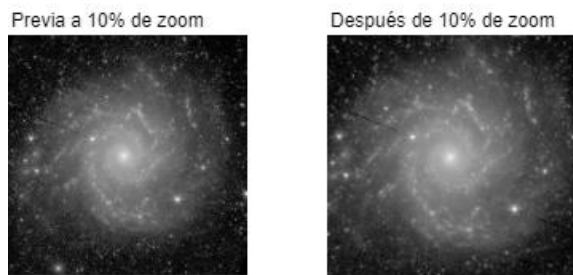


Figura 16 Zoom de galaxia espiral NGC628

El aplicarle estas distintas transformaciones al conjunto de datos nos asegura que habrá diversidad en todo el conjunto de datos debido a que el hecho de cambia posición, recortar, hacer *zoom*, etc., hace que la red aprenda características distintas de las imágenes independientemente de que sea la misma imagen.

Ciertas consideraciones tuvieron que hacerse para analizar, implementar y ejecutar este proceso de aumento de datos:

1. Las imágenes para transformarse deben seleccionarse de forma aleatoria.

2. Se puede aplicar de una a todas las transformaciones disponibles por imagen.
3. Las transformaciones deben seleccionarse de forma aleatoria.
4. Si una imagen llegase a seleccionarse para transformación más de una vez, las transformaciones nuevas a aplicar deberán ser distintas a cualquier combinación de transformaciones ya efectuadas a dicha imagen.

A continuación, el Algoritmo 1 es el encargado de generar las transformaciones aleatorias que se le aplicarán a las imágenes, un punto a considerar para este algoritmo es que la variable llamada TRANSFORMACIONES_DISPONIBLES es una lista que contiene las transformaciones que se pueden aplicar a las imágenes, las cuales se definieron previamente.

```

1  funcion generar_transformaciones(nombre_imagen, transformaciones_aplicadas)
2
3      transformaciones_a_aplicar = obtener_aleatorio(1, longitud(TRANSFORMACIONES_DISPONIBLES))
4
5      si transformaciones_a_aplicar == longitud(TRANSFORMACIONES_DISPONIBLES) y puede_usar_combinacion
6          retorna TRANSFORMACIONES_DISPONIBLES
7      fin si
8      otro
9          transformaciones_candidatas=[]
10         conteo_transformaciones=0
11
12         mientras conteo_transformaciones < transformaciones_a_aplicar
13             indice_transformacion = obtener_aleatorio(0, longitud(TRANSFORMACIONES_DISPONIBLES))
14             transformacion=TRANSFORMACIONES_DISPONIBLES[indice_transformacion]
15
16             si transformaciones_candidatas no incluye transformacion
17                 transformaciones_candidatas.agrega(transformacion)
18                 conteo_transformaciones += 1
19             fin si
20         fin mientras
21
22         si puede_usar_combinacion(nombre_imagen, transformaciones_aplicadas, transformaciones_candidatas)
23             retorna transformaciones_candidatas
24         fin si
25         otro
26             retorna generar_transformaciones(nombre_imagen, transformaciones_aplicadas)
27         fin otro
28     fin otro
29
30     finotro
31
32 finfuncion

```

Algoritmo 1 Generador de transformaciones

El Algoritmo 1 funciona básicamente de la siguiente manera:

1. Como parámetros recibe el nombre de la imagen en cuestión y una lista de todas las transformaciones que se han aplicado previamente, esta lista se utiliza para prevenir que a una imagen no se le aplique un set de transformaciones más de una vez.
2. Obtiene de forma aleatoria el número de transformaciones que se le aplicarán a la imagen.
Si el número de estas transformaciones es igual a la cantidad de transformaciones

disponibles y se valida que se puede utilizar esta combinación, entonces retorna que todas las transformaciones se le pueden aplicar a la imagen.

3. En caso contrario el algoritmo selecciona de forma aleatoria transformaciones individuales hasta llegar a la cantidad de transformaciones que se aplicarán. Si la combinación generada de forma aleatoria puede utilizarse, regresa la combinación. En caso de que la combinación no pueda usarse, se hace una llamada recursiva al mismo algoritmo para generar una combinación diferente hasta que lo encuentre.

Se decidió implementar este mecanismo de esta manera porque considerando que el set de datos es pequeño y muy disparejo en cantidad de información entre clases, sería más fácil implementar una solución que actúe de forma estocástica. De haber decidido usar un acercamiento determinista, por ejemplo, tomar N cantidad de imágenes y aplicar N transformaciones (donde N es un número definido al inicio del programa) el algoritmo hubiera quedado más complejo por cuestiones de paridad en la información, sumando que tendríamos un set de datos muy “parecido” porque todas las imágenes tendrían las mismas transformaciones aplicadas, restándole diversidad al conjunto de información.

Otro punto de suma importancia a revisar una vez que se tiene un set de información equivalente es el determinar el proceso de uniformidad que se le aplicará al conjunto de información. Toda la información que se use para entrenar una red neuronal convolucional debe de cumplir con las mismas características: resolución y canal de color.

En lo particular, estas imágenes vienen en resoluciones muy variadas, la mayoría con el mismo canal de color, pero no todas. Debido a que la resolución de estas imágenes es muy grande y el poder de procesamiento para la red neuronal es limitado, se optó por reducir cada imagen a una resolución de 100x100 píxeles y utilizar un canal de color gris. En el Algoritmo 2 se muestra la función utilizada para hacer el procesamiento de las imágenes.

```
1  funcion transformar_imagen(imagen)
2      imagen_escalagrises=cv2.cvtColor(imagen, COLOR_A_GRISES)
3      imagen_redimensionada=cv2.resize(imagen_escalagrises, (100, 100), INTER_LINEAR)
4
5      retorna numpy.array(imagen_redimensionada)
6  finfuncion
```

Algoritmo 2 Función para transformar imágenes

El Algoritmo 2 es muy sencillo, simplemente toma la imagen y le aplica las transformaciones necesarias que es cambiar el canal de color a gris y hacer las imágenes de 100x100. Este *script* se implementó en el lenguaje *Python*, haciendo uso de la librería llamada *OpenCV*.

Utilizando el acercamiento anterior se pudo hacer uniforme el conjunto de información: ahora cada clase (tipo de galaxia) cuenta con 150 elementos, por lo tanto, la red neuronal convolucional ya aprenderá de forma uniforme sobre todas las clases, aunque hay un problema: sigue siendo poca información. Esto sigue siendo un problema y utilizar acercamientos de implementación manual como el anterior, aunque resultó fructífero para balancear el conjunto de información, el extenderlo hubiera terminado siendo un proceso engorroso debido a que manejar el historial de transformaciones y limitarse a un determinado número de transformaciones hubiera posiblemente generado un conjunto de información desigual.

Para resolver este problema se investigó una clase provista por *Keras* llamada *ImageDataGenerator*, esta clase en su constructor recibe distintos parámetros que corresponden a las transformaciones que se le quieren realizar a las imágenes y se le pasa como argumento el directorio del conjunto de datos. Esta clase retorna un iterador que siempre estará transformando imágenes y regresándolas en batches para hacer uso de ellas. En este caso, se iteró sobre el objeto hasta que pudieron generarse 1,800 elementos de cada clase, conformando un conjunto de datos mucho más grande que el anterior. El Algoritmo 3 muestra la implementación de esta función.

```
1  generador=ImageDataGenerator(  
2      rotation_range=40,  
3      height_shift_range=0.2,  
4      zoom_range=0.2,  
5      horizontal_flip=True,  
6      fill_mode='constant'  
7  )  
8  
9  cantidad_en_batch=10  
10 directorio_conjunto_datos=Path('directorio')  
11  
12 iterador=generador.flow_from_directory(directorio,  
13     color_mode='grayscale',  
14     batch_size=cantidad_en_batch,  
15     class_mode='categorical'  
16 )  
17  
18 iteraciones=180  
19 inicio=0  
20 mientras imagen en iterador:  
21     inicio+=1  
22  
23     guardar_imagen(imagen)  
24  
25     si inicio == iteraciones:  
26         finmientras  
27  
28 finmientras
```

Algoritmo 3 Generador de imágenes de Keras

3.5 Diseño de autoencoder

Visto ya que un autoencoder no es más que una red neuronal donde lo que se presume es comprimir la información de entrada de tal manera que al descomprimirla, o sea, al reconstruirse, se obtenga algo lo más parecido posible a la información de entrada, se pensó en un modelo sencillo de bajo procesamiento pero que sea eficaz en su tarea.

El Algoritmo 4 pertenece al lenguaje Python, donde se utiliza la librería *Keras* para crear el autoencoder. Aquí muestra la arquitectura del autoencoder utilizado para las tres clases que se pretenden predecir.

```
1 input_layer = Input(shape=(100, 100, 1))
2 # Encoder
3 encoding_conv_layer_1 = Conv2D(75, (3, 3), activation='relu', padding='same')(input_layer)
4 encoding_pooling_layer_1 = MaxPooling2D((2, 2), padding='same')(encoding_conv_layer_1)
5 encoding_conv_layer_2 = Conv2D(50, (3, 3), activation='relu', padding='same')(encoding_pooling_layer_1)
6 encoding_pooling_layer_2 = MaxPooling2D((2, 2), padding='same')(encoding_conv_layer_2)
7 # Decoder
8 decoding_conv_layer_1 = Conv2D(50, (3, 3), activation='relu', padding='same')(encoding_pooling_layer_2)
9 decoding_upsampling_layer_1 = UpSampling2D((2, 2))(decoding_conv_layer_1)
10 decoding_conv_layer_2 = Conv2D(75, (3, 3), activation='relu', padding='same')(decoding_upsampling_layer_1)
11 decoding_upsampling_layer_2 = UpSampling2D((2, 2))(decoding_conv_layer_2)
12 output_layer = Conv2D(1, (3, 3), activation='sigmoid', padding='same')(decoding_upsampling_layer_2)
```

Algoritmo 4 Autoencoder

En el primer apartado que comienza con el *Encoder* y termina donde comienza el *Decoder*, se hace lo siguiente (línea por línea, omitiendo el comentario):

1. Se crea un objeto *Input* el cual genera una instancia de un tensor de *Keras*. En este objeto se pueden agregar ciertos atributos que permiten crear un modelo. El único argumento especificado por nosotros es *shape*, el cual es una tupla de dos enteros que indica la dimensionalidad de los vectores que se ingresarán, en este caso de 100x100 porque esa es la dimensión de nuestras imágenes.
2. Se genera un *kernel* de convolución, el cual se procesa con el *Input* produciendo un tensor de salidas. Existen múltiples argumentos que se pueden pasar a este objeto, pero solo se utilizaron los siguientes:
 - *Filters* (primer argumento): Representa la dimensionalidad del espacio de salida.
 - *Kernel size* (segundo argumento): Representa el tamaño de la ventana que se utilizará para realizar la convolución.
 - *Activation*: Representa la función de activación que se le aplicará al tensor de salidas.

- *Padding*: Representa el cómo pasará la ventana de convolución sobre la imagen original, el *padding same* es el que se usa de forma general puesto que la salida de la convolución tendría la misma dimensión que el original.
3. Esta tercer capa es crucial para el funcionamiento adecuado del autoencoder, *MaxPooling2D*. Esta capa es muy importante e interesante de aplicar debido a que hasta esta línea solo se ha trabajado con capas convolucionales, las cuales son buenas para obtener una síntesis de la presencia de ciertas características en las imágenes de entrada. Un problema que se tiene con esto es que los mapas de características se vuelven muy sensibles a la posición de las características en la información de entrada. La forma de atacar esto es reducir los mapas de características, haciendo que estos mismos sean más robustos ante los cambios de posición de las características. La capa *MaxPooling2D* toma solamente las características más activadas, es decir, las características que tienen más presencia en el mapa de características.

Como se puede observar, este proceso se repite una vez más. Es de notar que en la siguiente iteración se reducen los *filters* (dimensionalidad del espacio de salida) obligando a que el *encoder* seleccione solamente las características más importantes en cada instancia.

La siguiente sección del autoencoder, naturalmente, es el decoder. En esta sección del modelo lo que se pretende es “descomprimir” o “armar” lo codificado por el *encoder*, obteniendo solamente las características más importantes de la imagen, eliminando el ruido.

1. Se genera un *kernel* de convolución, el cual se inicializa con la información codificada y un número de filtros iguales a la última sección del *encoder*. Existen múltiples argumentos que se pueden pasar a este objeto, pero solo se utilizaron los siguientes:
- *Filters* (primer argumento): Representa la dimensionalidad del espacio de salida.
 - *Kernel size* (segundo argumento): Representa el tamaño de la ventana que se utilizará para realizar la convolución.
 - *Activation*: Representa la función de activación que se le aplicará al tensor de salidas.

- *Padding*: Representa el cómo pasará la ventana de convolución sobre la imagen original, el *padding same* es el que se usa de forma general puesto que la salida de la convolución tendría la misma dimensión que el original.
2. La siguiente línea, hace un proceso contrario al *MaxPooling2D*. Las capas *UpSampling2D* deben corresponder en parámetros a las *MaxPooling2D*, debido a que estas capas lo que hacen es muy sencillo, toman el mapa de características y lo repiten cuantas veces estén definidas en el parámetro *kernel size*.

De igual manera que en el encoder, esta operación se repite una vez más, pero, con una diferencia: en el decoder el número de filtros (la dimensionalidad) va aumentando en vez de disminuirse. Esto se debe al principio de reconstrucción que siguen los *decoders*.

Una vez teniendo el modelo se procede a guardar [21] el modelo de aprendizaje, para que este pueda ser utilizado en distintos entornos.

3.6 Diseño de red neuronal convolucional

Este apartado fue más complejo en su creación que el autoencoder puesto que se cuentan con más capas y hay más funciones de *Keras* involucradas, así como ciertos aspectos importantes a notar. Esta red neuronal convolucional es de tipo multiclase, es decir: una entrada (imagen) puede pertenecer a múltiples clases que son mutuamente excluyentes.

Más sencillo: esta red clasifica las galaxias de tal manera que te regresa las probabilidades de que las galaxias pertenezcan a cualquiera de las tres categorías, pero sin que una galaxia pueda pertenecer completamente a más de una categoría.

Es necesario recalcar que todas las imágenes que vayan a entrar a esta red neuronal necesitan pasar por el mismo procesamiento que las imágenes que pasaron por los autoencoders:

- Redimensión a 100x100.
- Cambio de canal de color a gris.

El Algoritmo 5 muestra la arquitectura de la red neuronal convolucional. Se omitió la función que transforma el input debido a que ya fue discutido anteriormente.

```
1  model = Sequential()
2
3  model.add(Conv2D(32, kernel_size=(3, 3), input_shape=input_shape))
4  model.add(Activation('relu'))
5  model.add(MaxPooling2D(pool_size=(2, 2)))
6
7  model.add(Conv2D(32, kernel_size=(3, 3)))
8  model.add(Activation('relu'))
9  model.add(MaxPooling2D(pool_size=(2, 2)))
10
11 model.add(Conv2D(64, kernel_size=(3, 3)))
12 model.add(Activation('relu'))
13 model.add(MaxPooling2D(pool_size=(2, 2)))
14
15 model.add(Flatten())
16 model.add(Dense(64))
17 model.add(Activation('relu'))
18 model.add(Dropout(0.5))
19 model.add(Dense(num_classes))
20 model.add(Activation('softmax'))
21 model.compile(loss='categorical_crossentropy', optimizer='rmsprop', metrics=['accuracy'])
22
23 train = model.fit(x_train, y_train,
24                  batch_size=batch_size,
25                  epochs=epochs,
26                  verbose=1,
27                  validation_data=(x_test, y_test),
28                  callbacks=[TensorBoard(histogram_freq=1)])
29
30 model.save()
```

Algoritmo 5 Arquitectura de red neuronal convolucional

La arquitectura de esta red es un modelo secuencial que consiste en tres iteraciones sobre el proceso: *Conv2D* => *ReLU* => *MaxPooling2D*. Finalizando con la siguiente secuencia: *Flatten* => *Dense* => *ReLU* => *Dropout* => *Dense* => *Softmax*. Como las capas *Conv2D* y *MaxPooling2D* ya fueron explicadas con anterioridad, se omitirá su explicación en la descripción de este modelo y se explicarán a detalle solamente los términos nuevos, como las funciones de activación y capas distintas a las detalladas anteriormente.

La primera capa nueva, *Flatten*, es una capa muy sencilla que siempre debe de ir antes de las capas completamente conectadas (*Dense* en este caso). Esta capa simplemente convierte el mapa de características actual (que en este caso viene de una capa *MaxPooling2D*) a un vector, que será lo que se le entrega a la capa completamente conectada.

La siguiente capa, *Dense*, es una capa completamente conectada, es decir, cada neurona en la capa recibe como entrada el resultado de las neuronas de la capa anterior. Estas capas son las indicadas a usarse en problemas de clasificación puesto que estas capas reciben el mapa de características

completo a diferencia de otras capas (ej. lineales). Esta capa puede estar presente en cualquier lugar de la arquitectura de la red neuronal, pero se debe de ajustar dependiendo del problema dado.

Las capas completamente conectadas pasan el resultado a través de la función de activación, la cual varía dependiendo del problema, en este caso como se puede apreciar se utiliza *ReLU* en primera instancia. Para problemas de múltiples clases como este, se recomienda la función de activación *softmax* puesto que esta función te regresa la probabilidad que tiene la entrada de pertenecer a cada clase a diferencia de otras funciones que solo te dicen si sí o no pertenece a la clase. El uso de la capa *Dense* se hace dos veces en el bloque de finalización, esto con el fin de reducir la dimensionalidad de la información poco a poco hasta llegar al número de clases que se busca clasificar. En estos problemas de clasificación, la capa densamente conectada es la última a definirse y su parámetro debe de ser siempre el número de clases que se intenta predecir para el input.

La siguiente capa, *Dropout* lidia con un problema llamado sobreajuste. Este problema en lo particular se refiere a cuando el resultado del entrenamiento de la red neuronal es muy parecido a los datos originales de entrada. Se lee como algo bueno, pero realmente el hecho de que una red neuronal tenga sobreajuste hace muy probable que la red neuronal falle o de “falsos positivos” con información diferente a la original. Es decir, hace que la red sea dependiente de la información con la que se entrenó.

Para hacer frente a este problema, una capa *Dropout* hace un proceso donde “desconecta” u omite neuronas durante el proceso de entrenamiento de la red neuronal, llevando a cabo un proceso llamado regularización. De esta manera se previene que las neuronas de la red hagan dependencia de la información de entrada. La Figura 17 muestra una red neuronal antes y después de hacer un *Dropout*.

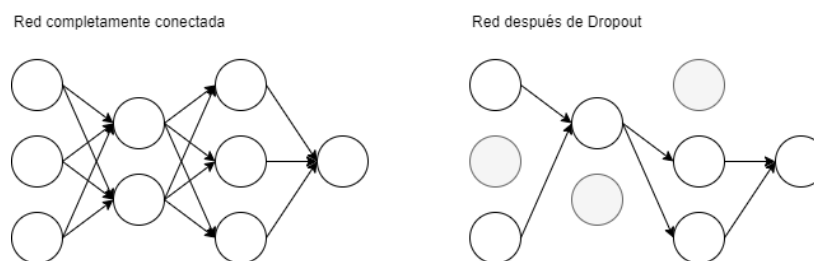


Figura 17 Aplicación de Dropout

Para finalizar, las redes neuronales convolucionales hacen uso de funciones de activación para calcular la suma de las entradas y las tendencias de estas entradas, el resultado de esta suma es utilizado para saber si una neurona se descartará o no, estas capas de activación se agregaron en varias partes del modelo. Existen múltiples funciones de activación e incluso pueden ser definidas por el usuario, pero a efectos de esta red neuronal convolucional se decidió ir por la función de activación ReLU (*Rectified Linear Activation Function*). *ReLU* es una función de activación que, como muchas otras, transforma la suma de los pesos de la entrada, convirtiéndolo en una salida. *ReLU* funciona de una manera muy simple: si la entrada recibida es de valor positivo, regresa ese valor y si es negativo, regresa un cero [22]. Esta función de activación es el default en una gran cantidad de tipos de redes neuronales debido a que sus operaciones matemáticas son muy sencillas, obteniendo un rendimiento alto.

3.7 Experimento de entrenamiento de red neuronal convolucional con imágenes sin aumento de datos y sin limpieza de ruido digital

La arquitectura anterior es simple pero potente, tiene una secuencia adecuada de convoluciones y filtros a aplicar. No es extremadamente pesada en procesamiento, pero cumple su función. Ahora antes de hacer la limpieza de imágenes y aplicar el procedimiento de eliminación del ruido digital se realizó un experimento con los datos originales, es decir, sin eliminarles el ruido digital y sin aumento de datos. Por lo tanto, en este experimento la red neuronal convolucional fue entrenada con un número subóptimo de datos, además de que las imágenes presentan alteraciones introducidas por el ruido digital. Recordemos que el conjunto original de datos está conformado de la siguiente manera:

- 18 imágenes de galaxias elípticas.
- 130 imágenes de galaxias espirales.
- 38 imágenes de galaxias lenticulares.

Las redes neuronales convolucionales al momento de efectuar el entrenamiento sobre la información requieren de ciertos parámetros con los cuales va a operar, uno de ellos es uno llamado *epochs*. Este parámetro representa el número de iteraciones que se harán sobre el conjunto de información. Inicialmente se dejó un número de 100 *epochs*. Inicialmente no se notó ningún tipo de problema o alteración con este acercamiento hasta que se revisó detalladamente el historial de

entrenamiento. Se descubrió que el número de *epochs* estaba haciendo que el modelo de la red neuronal convolucional memorizara el conjunto de datos de entrenamiento, es decir, estaba ocurriendo un sobreajuste. Se hicieron pruebas con números menores de *epochs* y se llegó a la conclusión de que un número de 25 *epochs* funciona de forma correcta en este modelo, a partir de este número se comienza a hacer sobreajuste sobre el conjunto de datos.

Otro punto importante para tratar dentro de los parámetros que reciben las redes neuronales convolucionales son las llamadas *loss functions*, estas funciones son utilizadas para medir que tan bueno es un modelo predictivo en relación con la información con la que se le alimentó, es decir, qué tan bueno es el modelo para predecir lo que se espera que prediga. Existen una infinidad de *loss functions*, pero hay ciertas funciones más adecuadas que otras para ciertas tareas, o funciones que fueron diseñadas para un dominio en específico. Hay que recordar que este modelo de redes neuronales convolucionales es multiclase, lo que significa que una imagen puede pertenecer como máximo a una clase, pero tiene probabilidad de pertenecer a alguna de las otras clases. Para efectos de esta red neuronal, se especificó *categorical cross entropy* como la *loss function* para este modelo. Esta función lo que hace es cuantificar la diferencia entre dos distribuciones de probabilidad, de esta forma el modelo aprende a asignarle una probabilidad alta a la clase que piensa que es correcta y una probabilidad baja a las demás clases del modelo.

Como argumento final a detallar, las redes neuronales convolucionales reciben una función optimizadora como parámetro que se utiliza en el momento del entrenamiento. Las funciones optimizadoras lo que hacen es “moldear” el estado del modelo a una forma más correcta, tomando en cuenta distintos parámetros de la red neuronal como la *loss function*, el número de *epochs*, número de *batches*, etc. Al igual que *loss functions*, también existe un mundo de funciones optimizadoras, generalmente cada una apta para distintos problemas. La función que se utilizó para esta red se llama *Root Mean Square Propagation* o *RMSprop*, esta función de activación tiene un muy buen desempeño con conjuntos de información pequeños, además de que ayuda a prevenir el sobreajuste.

A continuación, la Figura 18 muestra dos gráficas, una contiene la exactitud del modelo en el entrenamiento por *epochs*, y la otra muestra la pérdida o inexactitud del modelo por *epochs*.

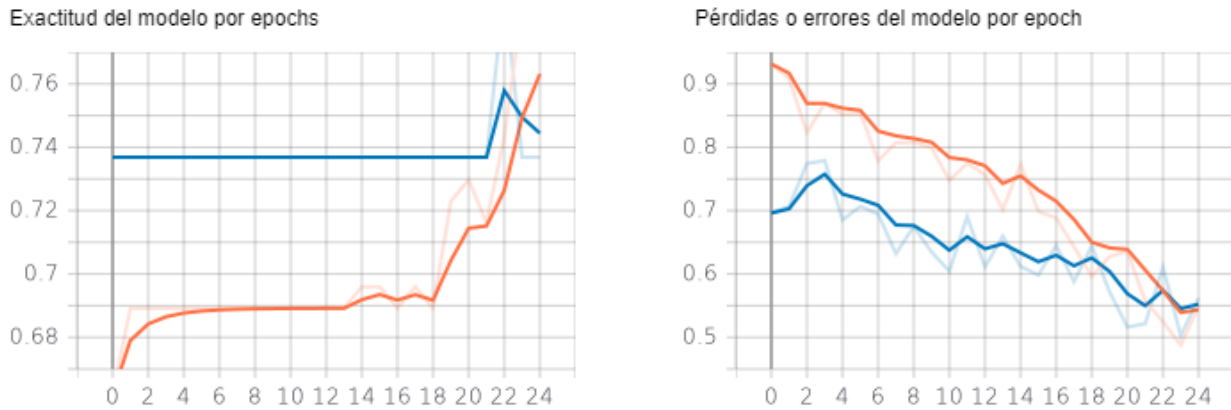


Figura 18 Primer experimento, exactitud e inexactitud del modelo

Recordando que esta red neuronal convolucional se entrenó en 25 *epochs* podemos ver que esta figura grafica el proceso de aprendizaje de la red a través de estas *epochs*. Las líneas de color naranja indican el aprendizaje (exactitud) en el entrenamiento y las líneas azules representan la pérdida (inexactitud) al validar el entrenamiento. Idealmente estas gráficas deberían de ser opuestas, es decir, mientras más aprenda el modelo menos inexactitud debería de tener. Cuando ocurre el sobreajuste estas gráficas no siguen ese patrón: un modelo que tiene sobreajuste sí aprende, pero aprende solo sobre los datos de entrenamiento y se desempeña pobremente en la validación, es decir, no tiene buen desempeño sobre datos no vistos puesto que se aprendió todo el conjunto de información en vez de sus generalidades.

Como podemos ver, este modelo no hizo un sobreajuste muy exagerado. La gráfica de pérdidas nos indica que las pérdidas en validación y entrenamiento son muy dispares, de igual forma ocurre con la gráfica de exactitud. Por comparación de estas gráficas se puede decir que el modelo no aprendió a generalizar bien las características del conjunto de información que se le entregó, teniendo un desempeño pobre.

En la Tabla 1 se muestra como quedaron los porcentajes finales en el entrenamiento de este primer experimento.

Tabla 1 Primer experimento, porcentajes finales de exactitud e inexactitud

	EXACTITUD	INEXACTITUD
ENTRENAMIENTO	76%	55%
VALIDACIÓN	74%	56%

El modelo terminó de entrenar con un porcentaje normal de aprendizaje en el conjunto de información de validación y entrenamiento, pero terminó con un porcentaje alto de inexactitud en entrenamiento y validación, lo que nos indica que esto no es óptimo.

3.8 Experimento de entrenamiento de red neuronal convolucional con imágenes después de aumento de datos sin limpieza de ruido digital

Dados los pobres resultados del entrenamiento anterior, se decidió hacer otro experimento. En cuanto a la arquitectura y parámetros de la red todo sigue igual, la diferencia ahora es que el conjunto de información es mucho mayor que el del experimento anterior. Ahora se tienen 1800 imágenes de cada categoría, por lo tanto, tenemos un conjunto de información balanceado.

La Figura 19 muestra las gráficas de exactitud e inexactitud del modelo a través de las *epochs*.

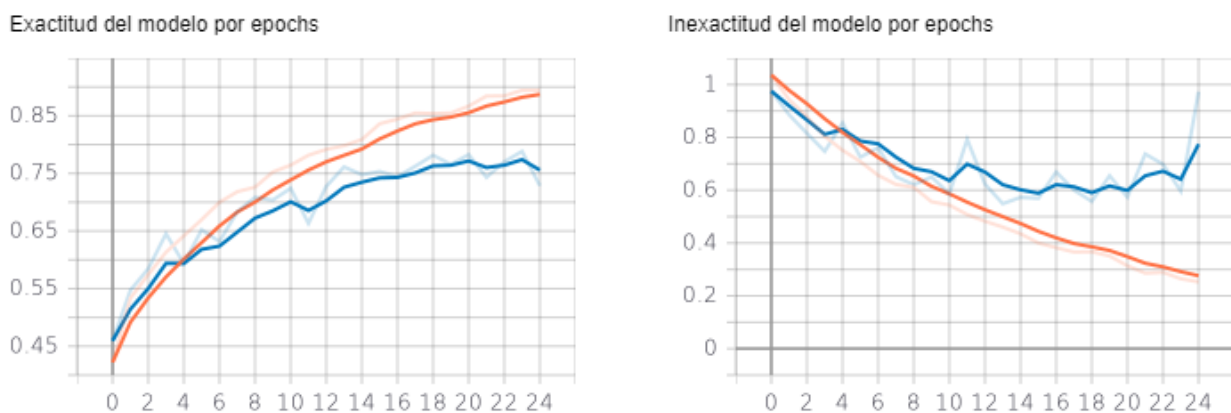


Figura 19 Segundo experimento, exactitud e inexactitud del modelo

Como podemos notar en estas gráficas y poniendo como contraste el experimento anterior es de notarse que tanto la exactitud e inexactitud de este segundo experimento fue un proceso mucho más gradual y se acomoda más a la naturaleza que debe de tener un modelo eficiente de redes neuronales convolucionales.

En la Tabla 2 se muestra como quedaron los porcentajes finales en el entrenamiento de este segundo experimento.

Tabla 2 Segundo experimento, porcentajes finales de exactitud e inexactitud

	EXACTITUD	INEXACTITUD
ENTRENAMIENTO	88%	27%
VALIDACIÓN	75%	77%

En este segundo experimento se incluyeron más ejemplos en el conjunto de información y el resultado resulta un poco más favorable durante el entrenamiento que durante la validación. El modelo sigue todavía generalizando de forma incorrecta, pero el resultado está mejorando.

3.9 Limpieza de imágenes

A continuación, en la Figura 20 se muestra una imagen original de la galaxia espiral NGC3351 y otra una vez que pasa por el autoencoder, después que sus características más importantes fueron seleccionadas.



Figura 20 NGC3351, antes y después de su limpieza

Es fácil notar que la imagen original es más oscura. Una vez que la imagen pasa por el autoencoder, este se encarga de seleccionar sus características más importantes, como podemos ver en la Figura 11, la forma de la galaxia está mucho más marcada y es más simple identificar a simple vista los bordes de este objeto astronómico.

Por otro lado, la imagen pasada por el autoencoder pierde un poco de nitidez, pero también pierde las características innecesarias como las estrellas demasiado brillantes o artefactos que pueden aparecer mediante la post edición o por interrupciones en la captura de la imagen. Se puede notar igual que la forma

de la galaxia se hace más notoria, comprobando que el autoencoder es de gran uso para conservar las características más importantes y remover las características innecesarias que pueden entorpecer el aprendizaje de la red neuronal convolucional.

Todas las imágenes del conjunto de información pasarán por este procedimiento de limpieza para eliminar las características menos importantes y contar con un conjunto de información limpio.

3.10 Entrenamiento de red neuronal convolucional con imágenes después de aumento de datos y después de limpieza de ruido digital

El último experimento consiste en el entrenamiento de la red neuronal convolucional ahora teniendo las imágenes libres de la mayor cantidad de ruido digital posible y con aumento de datos. Por lo tanto, se puede decir que este conjunto de información es de mejor calidad que los que se tenía en los experimentos anteriores. A continuación, la Figura 21 muestra las gráficas de exactitud e inexactitud a través de los *epochs*, de este tercer experimento.

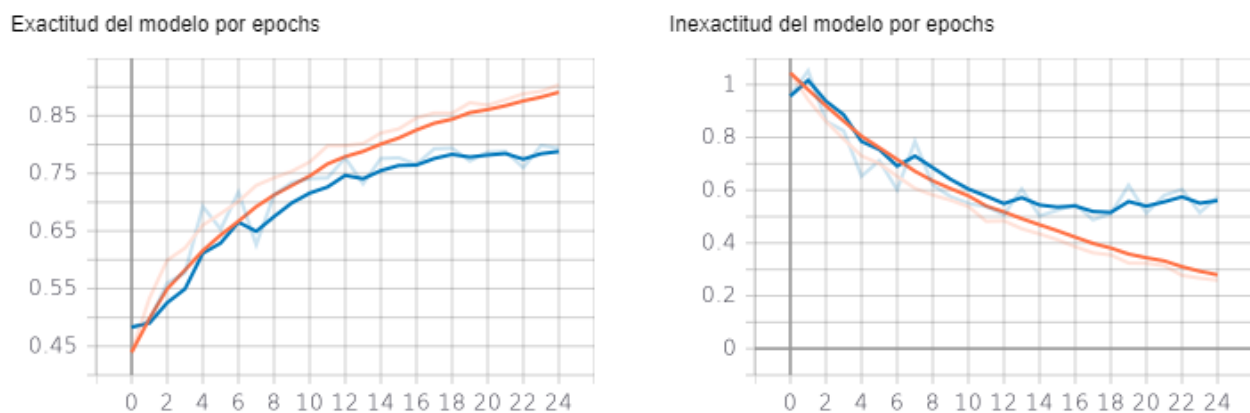


Figura 21 Tercer experimento, exactitud e inexactitud del modelo

Como se puede apreciar, esta gráfica se parece mucho al experimento anterior, con una gran diferencia en la exactitud del entrenamiento del modelo. En la Tabla 3 se muestran los porcentajes finales de este último experimento.

Tabla 3 Tercer experimento, porcentajes finales de exactitud e inexactitud

	EXACTITUD	INEXACTITUD
ENTRENAMIENTO	89%	27%
VALIDACIÓN	78%	56%

Es muy importante notar que, en este último experimento, mientras la exactitud se mantuvo casi en los mismos valores que en el experimento con el conjunto de información antes de su limpieza, la inexactitud en validación bajó aproximadamente un 20%, lo cual nos indica que el modelo con este conjunto de información generalizó mejor las características del conjunto de información que en los experimentos donde el conjunto de información no estaba limpio de ruido digital.

4 RESULTADOS

A lo largo de este proyecto de tesis se analizaron múltiples técnicas de inteligencia artificial para tareas de clasificación y manipulación de imágenes, específicamente redes neuronales convolucionales y *autoencoders*. Además de estas técnicas, se analizaron e implementaron distintas propuestas para aumentar el conjunto de información inicial, de tal forma que el conjunto pueda tener una cantidad equivalente de información a través de sus clases.

Tres experimentos fueron realizados con la información disponible y las distintas técnicas de inteligencia artificial presentadas aquí. De acuerdo con los resultados de los tres experimentos anteriores es posible decir con certeza que se respondió de forma positiva a la pregunta de investigación: sí es posible generar una red neuronal convolucional eficiente y confiable mediante un conjunto de información aumentado y con ruido digital reducido mediante otras redes neuronales.

La manera en que se llegó a este resultado fue de acuerdo con la forma en que se evaluó el desempeño de los distintos experimentos sobre el modelo de redes neuronales convolucionales, la forma de evaluación consistió en contrastar los porcentajes de exactitud y de inexactitud arrojados por el entrenamiento de los modelos en cada uno de los experimentos.

El primer experimento, donde el conjunto de información no ha sido limpiado ni aumentado, arrojó un aprendizaje de 76% en el aprendizaje, pero solamente un 54% de exactitud al momento de validar el entrenamiento, lo cual no es óptimo

En el segundo experimento el conjunto de información ha sido aumentado a un nivel donde todas sus clases son equivalentes, logrando un 88% de exactitud en el entrenamiento, pero solamente un 33% de exactitud al validar el entrenamiento lo cual es extremadamente bajo y nos demuestra que aun teniendo un conjunto de información equivalente, la calidad de este también influye.

Finalmente, en el último experimento el conjunto de información se encuentra en su estado final: clases equivalentes y con el ruido digital reducido por el *autoencoder*. El resultado de este experimento es parecido en su gráfica al primer experimento, arrojando los siguientes valores: 89% de exactitud en el entrenamiento y 44% de exactitud en la validación.

Por lo anterior se puede decir que un conjunto de información aumentado mediante técnicas de aumento de datos y procesado para eliminar el ruido digital con redes neuronales convolucionales sí permite generar un modelo predictivo eficiente.

5 CONCLUSIONES

Este proyecto se ha enfocado en la utilización de redes neuronales convolucionales para clasificar imágenes de galaxias, dichas imágenes pasando previamente por un proceso de limpieza de ruido digital mediante el uso de autoencoders, que son un tipo de red neuronal convolucional. Para cubrir esto se han desarrollado diferentes algoritmos que atacan los problemas presentados, desde la obtención y tratamiento de las imágenes, hasta la fase final donde se tiene un clasificador funcional.

El problema de mayor importancia del proyecto fue la limpieza de las imágenes de galaxias, esto se debe principalmente a que dichas imágenes provienen de telescopios, a los cuales sus sensores se mantienen en tiempos altos de exposición, necesarios para poder capturar las fotografías, sumándole que se aplica un post procesamiento con herramientas de diseño, aumentando el ruido existente. Entre los problemas presentados se encuentran: ruido digital en imágenes y poca cantidad de imágenes de galaxias.

Se logró superar ambos problemas presentados anteriormente, siendo posible aumentar la cantidad de información de imágenes de galaxias, conformando un set de datos equivalente, pudiendo hacer un entrenamiento más eficiente en la red neuronal convolucional. Con base en lo anterior, se realizó la prueba del sistema en tiempo real, donde se pudieron clasificar de manera exitosa imágenes de galaxias.

A continuación, un compilado de conclusiones puntuales:

- Tener un set de datos equivalente mejora de manera importante la precisión de una red neuronal convolucional, de igual manera reduciendo su pérdida.
- La limpieza de ruido de las imágenes antes de realizar el entrenamiento mejora de manera importante la eficiencia de la red neuronal convolucional, aumentando la precisión de la misma de manera considerable.
- La arquitectura de la red neurona convolucional es un punto muy importante, una arquitectura eficiente o probada previamente para problemas específicos será de mucha ayuda al momento de la resolución de este tipo de problemas.
- Por más que se aumente un conjunto de información, es preferente tener un conjunto de datos con una amplia cantidad de información en primera instancia. El aumento de datos si funciona para estos problemas donde no se tiene demasiada información apta para

utilizar, pero de igual forma que el aumento de datos trae beneficios, también puede ocasionar problemas. En este caso, el problema puede radicar en que el aumento de datos puede en cierta forma limitar la diversidad del conjunto de información.

6 COMPETENCIAS DESARROLLADAS

El conocimiento que adquirí durante el desarrollo de este proyecto fue principalmente en el área de inteligencia artificial y el ruido digital. Es muy interesante el saber e implementar otros usos que se les da a las redes neuronales convolucionales, en este caso, la reducción del ruido digital.

Entre las competencias que desarrollé se encuentra el análisis de las diferentes capas que tiene una red neuronal convolucional y que capas son más adecuadas para las tareas que se requieren realizar. Aprendí conceptos importantes y la razón de existencia del ruido digital. Mejoré en mis habilidades de creación de algoritmos optimizados y reusables, sobre todo en el aspecto de la transformación de imágenes. De igual manera mejoré bastante en mis habilidades de buscar e identificar información útil para el campo de este proyecto, habilidades que me servirán para otros proyectos futuros.

En conjunto con mis asesores logré definir el alcance del proyecto y fui capaz de realizar todas las tareas. Gracias a este proyecto se aporta un sistema que es capaz de clasificar imágenes de galaxias mediante el uso de redes neuronales convolucionales.

7 FUENTES DE INFORMACIÓN

1. M. A. C. Perryman, K. S. de Boer, G. Gilmore, E. Høg, M. G. Lattanzi, L. Lindegren, X. Luri, F. Mignard, O. Pace, P. T. de Zeeuw. (2001, abril 15). Numerical methods and codes. GAIA: Composition, formation and evolution of the Galaxy, 369, 363p.
2. R.B. Larson. (1992). Star Formation and Evolution. Galaxy Formation and Evolution. Cambridge University Press, 125p.
3. Beckwith, S., Stiavelli, M., Koekemoer, A., Caldwell, J.. (2007, diciembre). The Hubble Ultra Deep Field. The Astronomical Journal, 1729, 132p.
4. Weaver, D., Villard, R., Ilingworth, G., Andreoli G. (mayo, 2019). Hubble Astronomers Assemble Wide View of the Evolving Universe. NASA Goddard, de National Aeronautics and Space Administration Sitio web: <https://www.nasa.gov/feature/goddard/2019/hubble-astronomers-assemble-wide-view-of-the-evolving-universe>
5. Simon, H. (1995, agosto). Artificial intelligence: an empirical science. Artificial Intelligence, 77, 95-127pp.
6. McCarthy, J., Minsky, M., Rochester, N., Shannon, C.. (1955, agosto 31). A Proposal for the Dartmouth Summer Research Project on Artificial Intelligence. AI Magazine, 27, 15p.
7. Weizenbaum, J. (1996, enero). ELIZA—a computer program for the study of natural language communication between man and machine. Communications of the ACM, 9, 10p.
8. Yamasita, R. (2018). Convolutional neural networks: an overview and application in radiology. Insights into Imaging, 9, 611-629pp.
9. Baldi, P. (2011, julio). Autoencoders, unsupervised learning and deep architectures. UTLW 2011, 27, 37-50pp.
10. van der Maaten, L., Postma, E. (2009, octubre). Dimensionality Reduction: A Comparative Review. Tilburg centre for Creative Computing, Tilburg University.
11. Ying, X. (2019). An Overview of Overfitting and its Solutions. Journal of Physics, 1168, 2-3pp.
12. May, P. (2019, septiembre). Improved Image Augmentation for Convolutional Neural Networks by Copyout and CopyPairing. ArXiv.

13. European Space Agency. The Hubble tuning fork – classification of galaxies. 2020. European Space Agency, de European Space Agency. Sitio web <https://www.spacetelescope.org/images/heic9902o/>
14. Garner, R. Explore - The Night Sky | Hubble's Messier Catalog. Octubre 15, 2020, de National Aeronautics and Space Administration Sitio web: <https://www.nasa.gov/content/goddard/hubble-s-messier-catalog>
15. The Citizen Science Alliance. (2007). Galaxy Zoo. Obtenido de <https://www.galaxyzoo.org>
16. Brownlee, J. A Gentle Introduction to the Rectified Linear Unit (ReLU). Enero 9, 2019, de Jason Brownlee. Sitio web: <https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/>
17. LeCun, Y., Boser, B., Denker, J., Henderson, D., Howard R., Hubbard, W., Jackel, L. (1989, diciembre). Backpropagation Applied to Handwritten Zip Code Recognition. Neural Computation, 1, 541-551pp.