Proyecto 1: Machine Learning

Clasificación de imágenes de Mario Bross usando redes neuronales convolucionales

Daniel Mateo Guatibonza [*20161XXXX*] y Sofía Gutiérrez Rodríguez *[201612121]*

Ingeniería Electrónica  
Universidad de los Andes

1. ***Resumen*—Según Jeff Dean, líder de la rama de inteligencia artificial de Google, el número de proyectos que hacen uso de Deep Learning , es decir, la parte del machine learning que, por medio de algoritmos de alto nivel, imita la red neuronal del cerebro humano,[1] pasó de 0 a 1200 entre 2012 y 2016 [10]. Lo anterior puede deberse estos métodos han mejorado drásticamente el reconocimiento de habla, reconocimiento visual de objetos, su detección, entre otro[11]. Por lo tanto, el aumento en su uso puede deberse a la utilidad que tienen dichas redes en problemas de clasificación y en particular de imágenes.**
2. **Con el objetivo, de realizar una identificación de la existencia de este fenómeno se propone el uso de una red neuronal convolucional.**

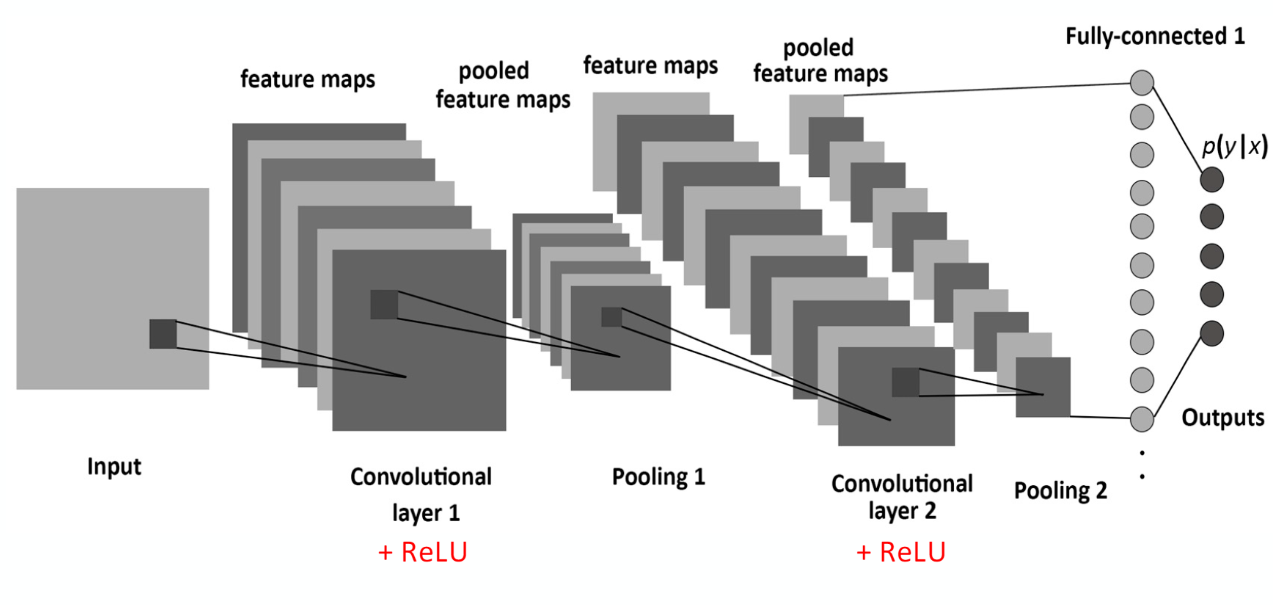
***Palabras clave—clasificación, exactitud, machine learning y red neuronal convolucional.***

Introducción

Redes Convoucionales

Una red neuronal es un modelo computacional inspirado en el comportamiento neuronal del cerebro humano. Este modelo se basa en la interconexión de nodos desde una capa de entrada, pasando por una serie de capas de nodos ocultos hasta llegar a la capa de salida que permite determinar relaciones entre neuronas [5]. Una red convolucional es una red neuronal donde las neuronas de determinada capa están conectadas solo a un grupo de neuronas de la capa anterior lo que permite explotar las correlaciones espaciales de determinada región, y por lo tanto son usadas generalmente para el procesamiento de imágenes [6].

La arquitectura general de una red convolucional se muestra a continuación:



Arquitectura de una red neuronal convolucional

En las capas convolucionales se reduce la dimensión extrayendo características por medio de filtros que codifican los pixeles. Para codificar los pixeles se hace el producto punto con una matriz de menor dimensión, la cual es llamada kernel o filtro. El resultado de esta operación son los mapas de activación, que indican las regiones donde características específicas del kernel fueron detectadas en la entrada [7]. Con función de activación RELU la cual está definida como max(0,x) y es la más utilizada en esta arquitectura.

Luego, generalmente hay capas de Pooling las cuales también reducen la dimensión perdiendo información con los propósitos de no sobrecargar computacionalmente las siguientes capas y evitar un sobre ajuste del modelo con los datos [7]. Y suelen consistir en escoger el máximo en las submatrices del tamaño definido

Posteriormente hay una capa completamente, a veces capas de dropout y finalmente la capa de salida, que en caso de ser un problema de clasificación la última capa al ser las salidas corresponde a las clases en las que se desea clasificar las entradas.

Identificación de Imágenes con Redes Convoucionales

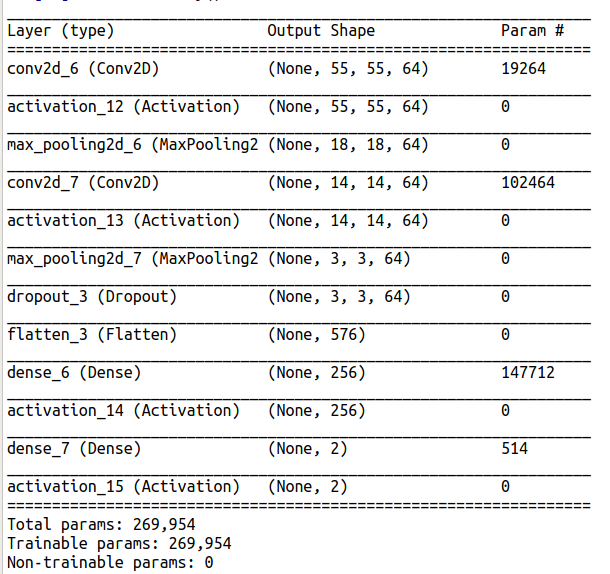
En primer lugar, cabe resaltar que los datos fueron entregados por Martínez y para probar la red neuronal se selecciona aleatoriamente el 10% de la los datos, del 90% restante se toma se toma el 80% para entrenar y el 20 para la validación del modelo. Al hacer esta partición de forma aleatoria se obtiene una diferente cada vez.

Luego, para la implementación de la red convolucional se toma como base la red propuesta por Yu-WeiChang para el reconocimiento de dígitos escritos a mano. Esta red tiene 2 bloques que consisten en 32 y 64 filtros de tamaño 5x5 respectivamente, y una activación relu en la primera capa, seguidos de una capa de pooling de 2x2. Posteriormente, una capa de dropout y para finalizar, una capa completamente conectada de 128 nodos con activación ‘softmax’, la cual al ser un problema de clasificación binaria se cambia por ‘sigmoid’. El número de parámetros y la forma de la salida se muestran por capa se muestran a continuación:

Resumen de la red neuronal convolucional

Con el fin de mejorar los resultados obtenidos, de acuerdo con los criterios que se mostrarán en la siguiente sección, se realizan las siguientes modificaciones:

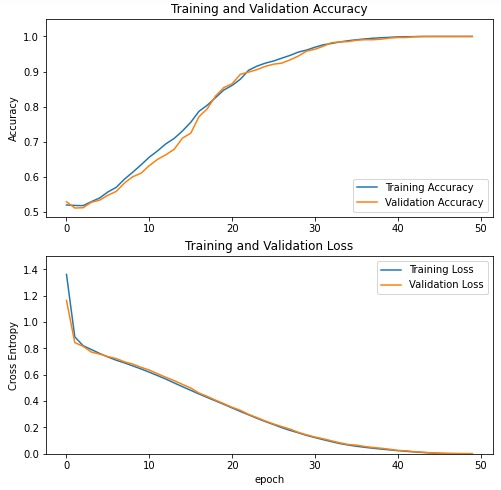
Ya que la red neuronal para reconocimiento de dígitos usa kernels de 5x5 para imágenes de 28x28 y según las recomendaciones de Rosebrock estos kernels se deberían usar para imágenes de 128x128 o más [8], entonces se decide aumentar el tamaño de kernel a 10x10 en la primera capa y mantener de 5x5 el de la segunda capa.  
Sin embargo, ya que al hacer esto, se reduce la dimensión y es importante no perder información relevante, se agregan 24 filtros más para extraer características inicialmente y se duplica el número de neuronas en la capa completamente conectada. Posteriormente se realiza sintonización fina manualmente para en el tamaño del pool hasta encontrar un mejor resultado obteniendo que los valores para la primera capa son 3x3 y para la segunda 4x4. Por lo que la red neuronal resultante en resumen queda de la siguiente forma:



Resumen de la red neuronal convolucional

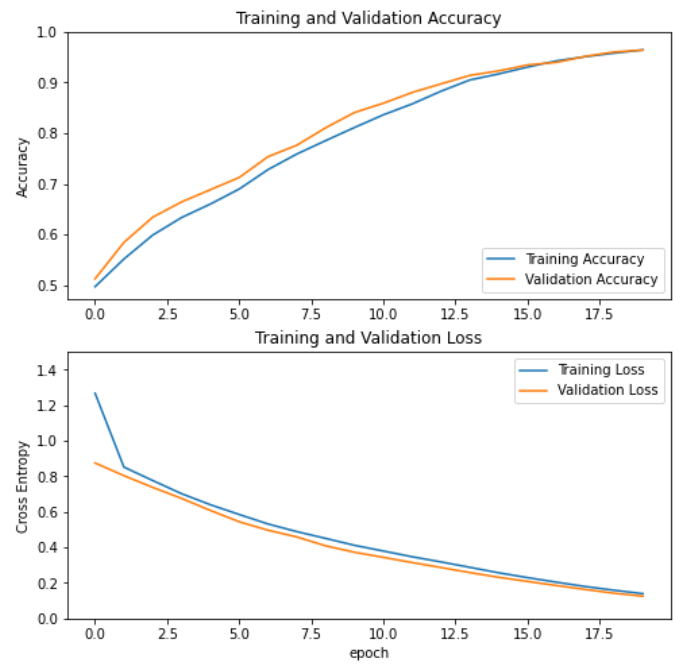
Resultados

Luego de tener el modelo es necesario saber cuántas épocas requiere el modelo para evitar sobre ajuste y que se logre un modelo con las características deseadas. Por lo tanto, se grafica la exactitud en función de las épocas para algunas particiones (6) y se puede ver que hay una tendencia de máximo en la época 8. Por lo tanto, se decide que éste será el número que se usará.

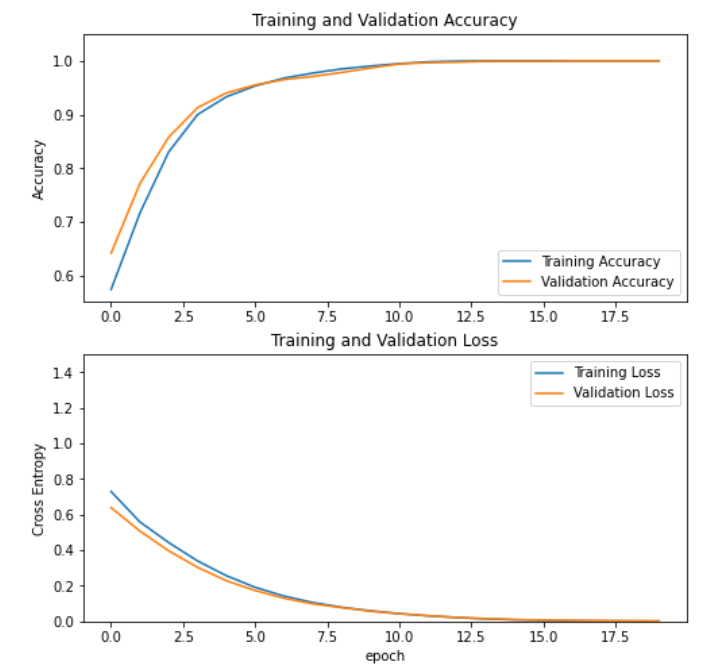


Exactitud en cada época

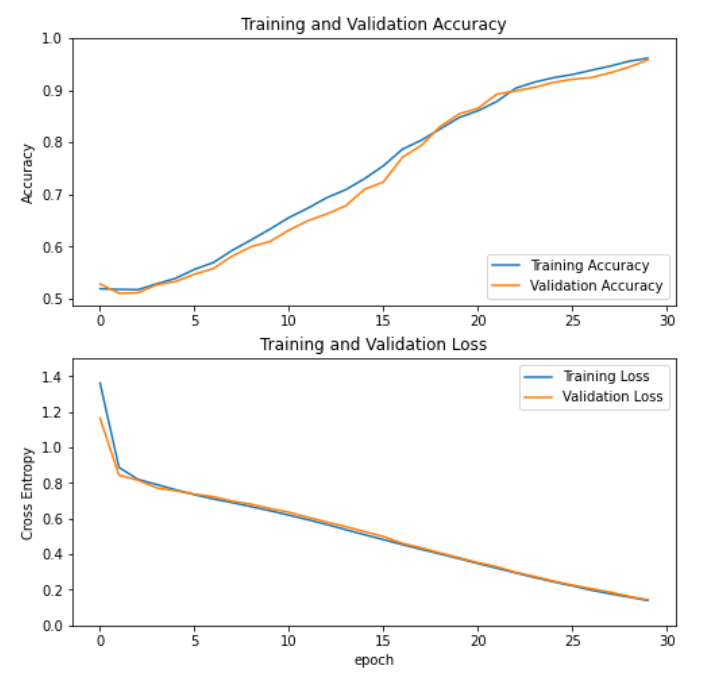
MUNDO 2 PEQUEÑA



MUNDO 6



Mundo 4/7



En primer lugar, para el caso de la red neuronal con activación ‘sigmoid’ en la última capa se obtiene la siguiente exactitud para las 10 particiones de los datos de entrenamiento, validación y prueba:

Exactitud para cada subconjunto de datos con la red convolucional inicial

|  |  |  |  |
| --- | --- | --- | --- |
| **Partición** | **Exactitud (Accuracy)** | | |
| ***Entrenamiento*** | ***Validación*** | ***Testeo*** |
| 1 | 0,681944444 | 0,758333333 | 0,825 |
| 2 | 0,697916667 | 0,808333331 | 0,85 |
| 3 | 0,6875 | 0,763888881 | 0,815 |
| 4 | 0,71875 | 0,819444452 | 0,8 |
| 5 | 0,748611111 | 0,49166666 | 0,58 |
| 6 | 0,688194444 | 0,808333335 | 0,805 |
| 7 | 0,655555556 | 0,830555554 | 0,865 |
| 8 | 0,715277778 | 0,822222228 | 0,86 |
| 9 | 0,767361111 | 0,597222221 | 0,7 |
| 10 | 0,694444444 | 0,525 | 0,45 |

Estadísticas De la Exactiud para la red convolucional inicial

|  |  |  |  |
| --- | --- | --- | --- |
| **Estadísticas** | ***Entrenamiento*** | ***Validación*** | ***Testeo*** |
| **Promedio** | 0,705555556 | 0,722499999 | 0,755 |
| **Desviación** | 0,009809028 | 0,156735344 | 0,17185 |

Se puede ver que este modelo tiene en una desviación estándar alta para los datos de testeo y validación, y en promedio una exactitud que se podría mejorar. Por lo tanto, a pesar de tener tendencias a exactitud, no se considera preciso y se modifica primeramente la función de activación de la salida por la función softmax y se obtienen los resultados que se muestran a continuación:

Exactitud para cada subconjunto de datos con la función de activación ‘softmax’

|  |  |  |  |
| --- | --- | --- | --- |
| **Partición** | **Precisión** | | |
| ***Entrenamiento*** | ***Validación*** | ***Testeo*** |
| 1 | 0,672222222 | 0,483333327 | 0,56 |
| 2 | 0,745138889 | 0,563888884 | 0,63 |
| 3 | 0,755555556 | 0,744444443 | 0,79 |
| 4 | 0,697222222 | 0,50555556 | 0,61 |
| 5 | 0,715972222 | 0,808333335 | 0,855 |
| 6 | 0,722916667 | 0.59999999 | 0,69 |
| 7 | 0,670833333 | 0,844444443 | 0,86 |
| 8 | 0,679166667 | 0,841666669 | 0,88 |
| 9 | 0,725694444 | 0,833333328 | 0,865 |
| 10 | 0,725694444 | 0,861111118 | 0,89 |

Estadísticas Del la Exactiud para la red convolucional modificada

|  |  |  |  |
| --- | --- | --- | --- |
| **Estadísticas** | ***Entrenamiento*** | ***Validación*** | ***Testeo*** |
| **Promedio** | 0,711041667 | 0,720679012 | 0,763 |
| **Desviación** | 0,008069493 | 0,197810361 | 0,14646 |

Se puede ver entonces que este cambio incrementa el promedio de la exactitud y lo hace más preciso. Esto se puede evidenciar gráficamente en (7 y 8) ya que para la segunda las líneas llegan a puntos más cerca y con valores superiores en la última época

Exactitud en cada época para sigmoid

Exactitud en cada época para sofmax

Sin embargo, la desviación sigue siendo alta y por lo tanto se hacen las modificaciones descritas anteriormente. Logrando que porcentajes superiores de exactitud y reduciendo la desviación aproximada en una cuarta parte, es decir mejorando radicalmente la precisión como se muestra en las siguientes tablas:

Exactitud para cada subconjunto de datos con la red neuronal final

|  |  |  |  |
| --- | --- | --- | --- |
| **Partición** | **Exactitud** | | |
| ***Entrenamiento*** | ***Validación*** | ***Prueba*** |
| 1 | 0,73125 | 0,75357145 | 0,8283333 |
| 2 | 0,7732143 | 0,60714287 | 0,70666665 |
| 3 | 0,77410716 | 0,86071426 | 0,86833334 |
| 4 | 0,73660713 | 0,76785713 | 0,83166665 |
| 5 | 0,74642855 | 0,80714285 | 0,7866667 |
| 6 | 0,78482145 | 0,7607143 | 0,84 |
| 7 | 0,71339285 | 0,8464286 | 0,865 |
| 8 | 0,73303574 | 0,7821429 | 0,85 |
| 9 | 0,73482144 | 0,85 | 0,85833335 |
| 10 | 0,7580357 | 0,85 | 0,85833335 |

Estadísticas Del la Exactiud para la red convolucional final

|  |  |  |  |
| --- | --- | --- | --- |
| **Estadísticas** | ***Entrenamiento*** | ***Validación*** | ***Prueba*** |
| **Promedio** | 0,748571432 | 0,788571436 | 0,829333334 |
| **Desviación** | 0,004778702 | 0,051831625 | 0,021890004 |

Además, otro logro representativo con respecto a las anteriores redes es que se logran curvas más suaves (8 y 9), lo que permitiría tener una predicción comportamiento del modelo a través de las épocas

Pérdidas en cada época para red neuronal final

Exactitud en cada época para red neuronal final

Análisis de Resultados

Al observar los datos mostrados en las tablas II, IV vs VI se evidencia que no hay comportamiento de trade-off entre la desviación y precisión, ya que ambos aumentan sin importar que este cambo se mucho más significativo en uno que en otro. Cabe resaltar que es una pequeña variación del fenómeno esta discutido en la literatura como el trade-off entre *bias y varianza.* Ya que el que haya errores de un orden de magnitud implica que no logra distinguir del todo las características más relevantes para poder diferenciar de manera acertada, es decir tiene un alto bias o poca exactitud. Mientras que la varianza está ligada a la desviación, e implica que tiene un modelo muy complejo tratando de ajustar un pequeño número de características; se dice entonces que el modelo tiene un sobre ajuste a los datos de entrenamiento y se aprende los datos de memoria y pierde la capacidad de generalizar el modelo para nuevos datos; es decir, entre más parecidos sean los datos a los de entrenamiento mejor desempeño tendrá. Por lo tanto, para lograr mejores soluciones los factores de reducción de dimensión afectivamente ayudaron a disminuir la complejidad del modelo generado. Sin embargo, se configuraron los parámetros adecuados para evitar pasar al punto en el que deja de reconocer las características importantes.

Lo anterior se debe también a la escogencia de los datos de entrenamiento, por lo que se podría decir para los casos en que el modelo era muy simplificado, o en otras palabras había alto bias, era debido a que la escogencia aleatoria los datos casualmente eran similares entre si. Por lo tanto, se podría aumentar la cantidad de datos de entrenamiento si es posible con datos nuevos y si no agregar un generador de imágenes que agregue las mismas imágenes modificadas y encontrar los filtros que mejor extraigan las características buscadas.

También, se observó que la mayoría de las veces el porcentaje obtenido en entrenamiento era menor al de validación y prueba, lo cual podría indicar que talvez se necesitaba un modelo más complejo ya que como se entrenaba con muchos datos este no alcanzaba a poder predecir todos los datos de entrenamiento pero si justo un subset de estos que son similares a los que logra predecir y por tanto se obtenían mejores resultados

Debido a que los resultados se ven influenciados por los datos. Se realiza una última prueba usando random Forest y como criterio la exactitud. Sin embargo, a pesar de que se reduce la desviación a 0,00599 también se reduce el promedio al 0,506. Esto se debe a que no hubo un proceso de caracterización, sino que se convierte la imagen directamente en una matriz, lo cual como se puede ver no es muy recomendable, aunque para algunos casos como para el de [9] podría funcionar mejor.

Conclusiones

Se pudo evidenciar la utilidad del uso de una red convolucional para un problema de astrofísica, además, cabe resaltar que cada vez son más los campos en los que se requiere del procesamiento de imágenes y que cada vez más lo datos que se tienen. Para esto se puede usar machine learning, sin embargo, es necesario identificar el problema si es de clasificación, identificación de objetos dentro de una imagen o clusterización, para utilizar la herramienta requerida.

En cuanto a los parámetros e hiperparámetros, se mostró importante conocer el significado en el modelo de cada uno y su campo de acción con el fin de obtener un mejor modelo. Como por ejemplo el número de filtros, su tamaño, el número de neuronas, entre otros. También es necesario realizar ciertas visualizaciones de los datos ya que estas dan idea de ciertos parámetros como el número de épocas para evitar sobre ajuste en caso de que en la gráfica de la figura 7 las curvas fueran naranja validación y azul entrenamiento, por ejemplo.

Por otro lado, se evidencio la importancia de las bases de datos y la procuración a tener una base de datos amplia y además escoger con mayor robustez los datos de entramiento y prueba. Además, también es necesario tener en cuenta la capacidad computacional, ya que si esta de base de datos o el modelo son muy grandes, debería ser corrido en una GPU.

Finalmente, las posibles mejoras a futuro consistirían en realizar una búsqueda de filtros que mejoren la extracción de características permitiendo así un menor bias y al implementar tanto una partición más robusta de los datos y una base datos más completa, se reduciría la varianza y se suavizarían las curvas de la exactitud por época. O en caso de que estos no funcionen como deberían una nueva búsqueda de parámetros e hiper parámetros pero desde una GPU para agilizar el proceso.

Por otro lado, como trabajo adicional y de contraste realizar el clasificador con el algoritmo de random forest extrayendo las características por medio de un PCA para comparar con los resultados obtenidos por la red convolucional mejorada.

Referencias

SalesForce. 2018. “Machine Learning y Deep Learning: aprende las diferencias”. Disponible en: <https://www.salesforce.com/mx/blog/2018/7/Machine-Learning-y-Deep-Learning-aprende-las-diferencias.html>

E. Eiroa. “Estudios sobre lentes gravitacionales con objetos relativistas”. Universidad de Buenos Aires, Facultad de Ciencias Exactas y Naturales Departamento. Disponible en <https://digital.bl.fcen.uba.ar/download/tesis/tesis_n3448_Eiroa.pdf>

T. Aguirre. “Lentes gravitacionales fuertes: análisis y detección” Disponible en <https://rdu.unc.edu.ar/bitstream/handle/11086/6247/trabajoTagliaferro.pdf?sequence=1&isAllowed=y>

Instituo de Astrofísica de las canarias. “Lentes Gravitacionales”. Disponible en <https://www.iac.es/cosmoeduca/gravedad/fisica/fisica1.htm>

SARKAR D, BALI R., SHARMA T. 2018 Practical Machine Learning

with Python Apress, India.

R. Martinez y P. Protopapas. “Introduction to convolutional networks”. Harvard-Smithsonian Cener for Astrophisics

A. Saxena. 2016. “Convolutional Neural Networks (CNNs): An Illustrated Explanation”. Disponible en\_<https://blog.xrds.acm.org/2016/06/convolutional-neural-networks-cnns-illustrated-explanation/>

A. Rosebrock. “Keras Conv2D y capas convolucionales”. 2018. Disponible en línea en: <https://www.pyimagesearch.com/2018/12/31/keras-conv2d-and-convolutional-layers/>

Hyperion Development. 2019. “Python machine learning: Introduction to image classification”. Disponible en <https://blog.hyperiondev.com/index.php/2019/02/18/machine-learning/>

J. Dean. “Large-Scale Deep Learning for Intelligent Computer Systems”. Disponible en <http://www.wsdm-conference.org/2016/slides/WSDM2016-Jeff-Dean.pdf>

Y. LeCun, Y. Bengio y G. Hinton. . 2015 . “Deep Learning”. Disponible online en: <https://doi.org/10.1038/nature14539>

 Stanford CS class. Sf. “Convolutional Neural Networks (CNNs / ConvNets)“ Disponible en linea en http://cs231n.github.io/convolutional-networks/#overview

1. ***Resumen*—Según Jeff Dean, líder de la rama de inteligencia artificial de Google, el número de proyectos que hacen uso de Deep Learning , es decir, la parte del machine learning que, por medio de algoritmos de alto nivel, imita la red neuronal del cerebro humano,[1] pasó de 0 a 1200 entre 2012 y 2016 [10]. Lo anterior puede deberse estos métodos han mejorado drásticamente el reconocimiento de habla, reconocimiento visual de objetos, su detección, entre otro[11]. Por lo tanto, el aumento en su uso puede deberse a la utilidad que tienen dichas redes en problemas de clasificación y en particular de imágenes.**
2. **Con el objetivo, de realizar una identificación de la existencia de este fenómeno se propone el uso de una red neuronal convolucional.**

***Palabras clave—clasificación, exactitud, machine learning, red neuronal convolucional.***

Introducción

El miércoles 19 de abril de este año se tomó la primera foto a un agujero negro, un logro notable para varias disciplinas, entre ellas la astrofísica, ciencia que estudia los astros, sus propiedades, su origen y evolución. Sin embargo, este logro no sería posible sin tener el previo conocimiento de donde se encuentra dicho agujero. Una manera de identificar la presencia de un agujero negro es identificando el lente gravitacional generado por este [2]. Por lo anterior, la distorsión generada por estos y otras razones que se explicarán en la siguiente sección de este informe, este fenómeno requiere un método de detección.

COSMOS, SLACS y el Programa Hubble Frontiers Field realizaron algoritmos para la detección de este fenómeno, además de la construcción de bases de datos a partir de este trabajo. Brevemente las metodologías utilizadas fueron: Los dos primeros trabajos, a grandes rasgos, realizan la substracción de imágenes y analizar los residuos de estas por medio de inspección visual. Por último, el Programa Hubble Frontiers Field utiliza un modelado matemático de una agrupación, para luego encontrar una predicción particular y compararla [3].

Por otro lado, otro método de detección previamente utilizado consistió en sustraer galaxias desde imágenes utilizando el algoritmo Análisis de Componente Principal, PCA, para restarla siguiendo el procedimiento similar al de de COSMOS y SLACS. Sin embargo, al final utilizan dos algoritmos para la detección. En el primero, lo convierten en un problema de detección de líneas al transformarlo a coordenadas polares. En el segundo extraen la orientación, multiplicidad y elongación para identificar la estructura. [3]

Finalmente, para el caso particular de los lentes gravitacionales fuertes, Aguirre propone como solución algoritmos de aprendizaje automático (machine learning) para el análisis de imágenes, tomando como referencia las bases de datos: COSMOS y el Programa Hubble Frontiers Field. Los algoritmos consisten en clasificadores binarios, donde las clases son es un arco gravitacional o no. Para esto, en primer lugar, extrae las características de las imágenes, por medio del código abierto Wndchrm, el cual extrae las características de las imágenes y de las Transformadas Rápidas de Fourier (FFT), wavelets, transformada de Chebyshev y la transformada de Edge. Posteriormente, la clasificación se realiza mediante los algoritmos de: naive Bayes, regresión logística y ramdom forest. Sin entrar en detalles, estos algoritmos consisten en la creación de un modelo a partir de las características y probabilidades. Para el primer caso el modelo es construido a partir teorema de Bayes, maximizando las probabilidades; mientras que, para el segundo, se busca maximizar la función logística y para último se usan arboles de decisión [3].

Lentes gravitacionales

De acuerdo con la teoría de la relatividad general de Einstein, la luz se curva cerca de una masa que deforma el espacio-tiempo a su alrededor y desvía la luz de otras galaxias lejanas. En particular, cuando estas masas son de tamaño considerable, este efecto es conocido como *lente gravitacional*, ya que al igual que los lentes estos deforman y/o amplifican la imagen de las galaxias lejanas produciendo imágenes dobles o múltiples, arcos, etc. Las deformaciones causadas pueden hacer que un cuásar parezca una galaxia en espiral [4] o que una galaxia u otro astro está más cerca de lo que realmente está, es decir pueden funcionar como telescopios naturales [3].

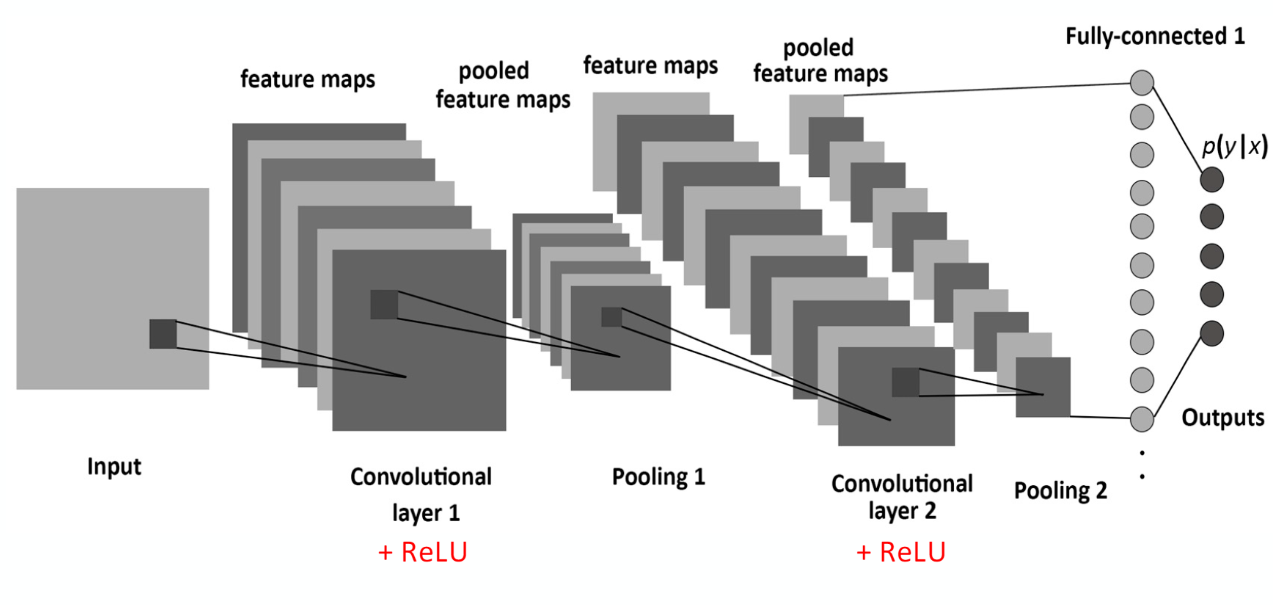
Entre los astros de masa considerable se encuentran agujeros negros, núcleos galácticos activos, agujeros de gusano [2]. Por lo que las lentes gravitacionales permiten su identificación y además sirven como herramienta para estudiar la distribución de materia oscura de una galaxia masiva, de cúmulos de galaxias [3].

Tanto para identificar la presencia de un agujero negro o para identificar si el objeto observado es realmente una galaxia o no, o si esta está realmente a la distancia estimada; es necesario identificar en primer lugar la presencia de un lente gravitacional. Teniendo en cuenta que los métodos anteriormente descritos requieren tiempo ya sea para el análisis visual o la previa extracción de características. Se propone el uso de una red convolucional.

Redes Convoucionales

Una red neuronal es un modelo computacional inspirado en el comportamiento neuronal del cerebro humano. Este modelo se basa en la interconexión de nodos desde una capa de entrada, pasando por una serie de capas de nodos ocultos hasta llegar a la capa de salida que permite determinar relaciones entre neuronas [5]. Una red convolucional es una red neuronal donde las neuronas de determinada capa están conectadas solo a un grupo de neuronas de la capa anterior lo que permite explotar las correlaciones espaciales de determinada región, y por lo tanto son usadas generalmente para el procesamiento de imágenes [6].

La arquitectura general de una red convolucional se muestra a continuación:



Arquitectura de una red neuronal convolucional

En las capas convolucionales se reduce la dimensión extrayendo características por medio de filtros que codifican los pixeles. Para codificar los pixeles se hace el producto punto con una matriz de menor dimensión, la cual es llamada kernel o filtro. El resultado de esta operación son los mapas de activación, que indican las regiones donde características específicas del kernel fueron detectadas en la entrada [7]. Con función de activación RELU la cual está definida como max(0,x) y es la más utilizada en esta arquitectura.

Luego, generalmente hay capas de Pooling las cuales también reducen la dimensión perdiendo información con los propósitos de no sobrecargar computacionalmente las siguientes capas y evitar un sobre ajuste del modelo con los datos [7]. Y suelen consistir en escoger el máximo en las submatrices del tamaño definido

Posteriormente hay una capa completamente, a veces capas de dropout y finalmente la capa de salida, que en caso de ser un problema de clasificación la última capa al ser las salidas corresponde a las clases en las que se desea clasificar las entradas.

Identificación de Lentes Gravitacionales con Redes Convoucionales

Ahora bien, retomando el caso particular del reconocimiento de lentes gravitacionales, con imágenes tan similares como (2) y (3), se hace evidente la necesidad del uso de una red convolucional.

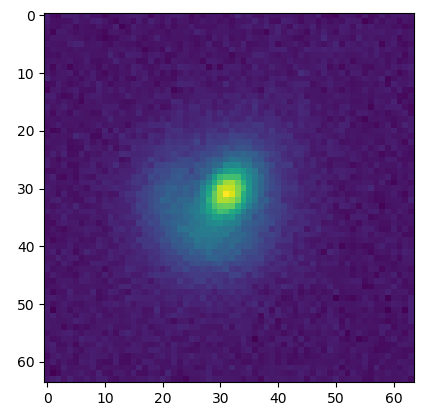


Imagen astronómica con presencia de lente gravitacional

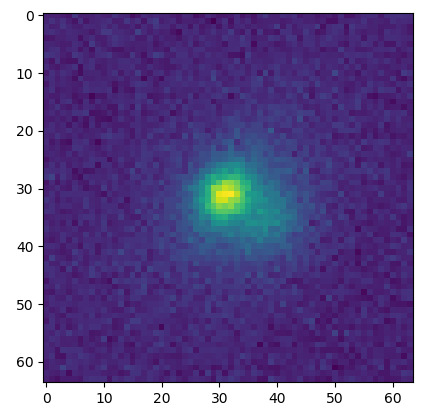
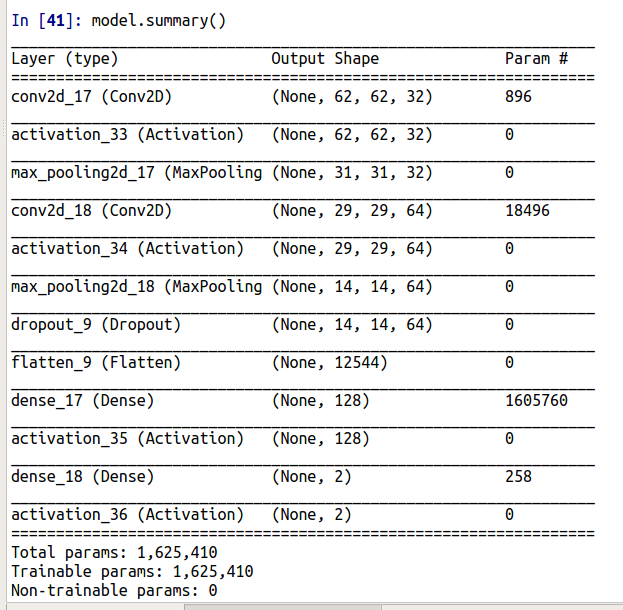


Imagen astronómica sin presencia de lente gravitacional

En primer lugar, cabe resaltar que los datos fueron entregados por Martínez y para probar la red neuronal se selecciona aleatoriamente el 10% de la los datos, del 90% restante se toma se toma el 80% para entrenar y el 20 para la validación del modelo. Al hacer esta partición de forma aleatoria se obtiene una diferente cada vez.

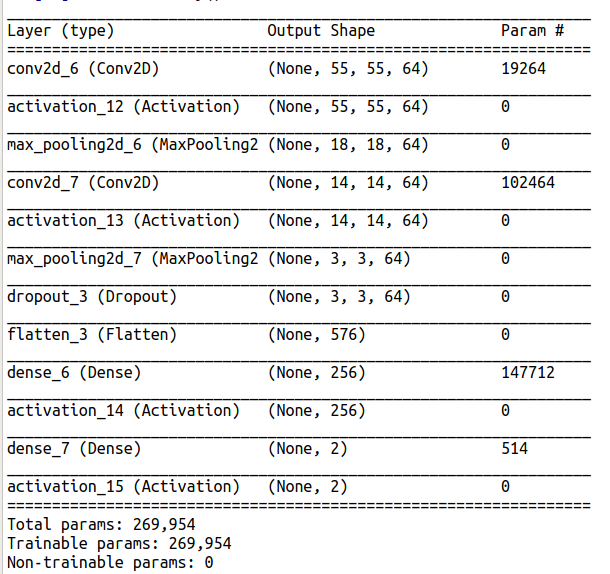
Luego, para la implementación de la red convolucional se toma como base la red propuesta por Yu-WeiChang para el reconocimiento de dígitos escritos a mano. Esta red tiene 2 bloques que consisten en 32 y 64 filtros de tamaño 5x5 respectivamente, y una activación relu en la primera capa, seguidos de una capa de pooling de 2x2. Posteriormente, una capa de dropout y para finalizar, una capa completamente conectada de 128 nodos con activación ‘softmax’, la cual al ser un problema de clasificación binaria se cambia por ‘sigmoid’. El número de parámetros y la forma de la salida se muestran por capa se muestran a continuación:



Resumen de la red neuronal convolucional

Con el fin de mejorar los resultados obtenidos, de acuerdo con los criterios que se mostrarán en la siguiente sección, se realizan las siguientes modificaciones:

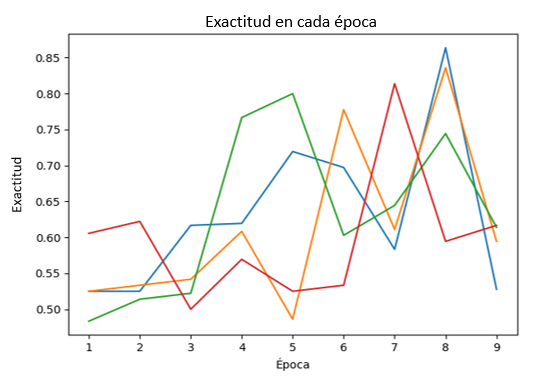
Ya que la red neuronal para reconocimiento de dígitos usa kernels de 5x5 para imágenes de 28x28 y según las recomendaciones de Rosebrock estos kernels se deberían usar para imágenes de 128x128 o más [8], entonces se decide aumentar el tamaño de kernel a 10x10 en la primera capa y mantener de 5x5 el de la segunda capa.  
Sin embargo, ya que al hacer esto, se reduce la dimensión y es importante no perder información relevante, se agregan 24 filtros más para extraer características inicialmente y se duplica el número de neuronas en la capa completamente conectada. Posteriormente se realiza sintonización fina manualmente para en el tamaño del pool hasta encontrar un mejor resultado obteniendo que los valores para la primera capa son 3x3 y para la segunda 4x4. Por lo que la red neuronal resultante en resumen queda de la siguiente forma:



Resumen de la red neuronal convolucional

Resultados

Luego de tener el modelo es necesario saber cuántas épocas requiere el modelo para evitar sobre ajuste y que se logre un modelo con las características deseadas. Por lo tanto, se grafica la exactitud en función de las épocas para algunas particiones (6) y se puede ver que hay una tendencia de máximo en la época 8. Por lo tanto, se decide que éste será el número que se usará.



Exactitud en cada época

En primer lugar, para el caso de la red neuronal con activación ‘sigmoid’ en la última capa se obtiene la siguiente exactitud para las 10 particiones de los datos de entrenamiento, validación y prueba:

Exactitud para cada subconjunto de datos con la red convolucional inicial

|  |  |  |  |
| --- | --- | --- | --- |
| **Partición** | **Exactitud (Accuracy)** | | |
| ***Entrenamiento*** | ***Validación*** | ***Testeo*** |
| 1 | 0,681944444 | 0,758333333 | 0,825 |
| 2 | 0,697916667 | 0,808333331 | 0,85 |
| 3 | 0,6875 | 0,763888881 | 0,815 |
| 4 | 0,71875 | 0,819444452 | 0,8 |
| 5 | 0,748611111 | 0,49166666 | 0,58 |
| 6 | 0,688194444 | 0,808333335 | 0,805 |
| 7 | 0,655555556 | 0,830555554 | 0,865 |
| 8 | 0,715277778 | 0,822222228 | 0,86 |
| 9 | 0,767361111 | 0,597222221 | 0,7 |
| 10 | 0,694444444 | 0,525 | 0,45 |

Estadísticas De la Exactiud para la red convolucional inicial

|  |  |  |  |
| --- | --- | --- | --- |
| **Estadísticas** | ***Entrenamiento*** | ***Validación*** | ***Testeo*** |
| **Promedio** | 0,705555556 | 0,722499999 | 0,755 |
| **Desviación** | 0,009809028 | 0,156735344 | 0,17185 |

Se puede ver que este modelo tiene en una desviación estándar alta para los datos de testeo y validación, y en promedio una exactitud que se podría mejorar. Por lo tanto, a pesar de tener tendencias a exactitud, no se considera preciso y se modifica primeramente la función de activación de la salida por la función softmax y se obtienen los resultados que se muestran a continuación:

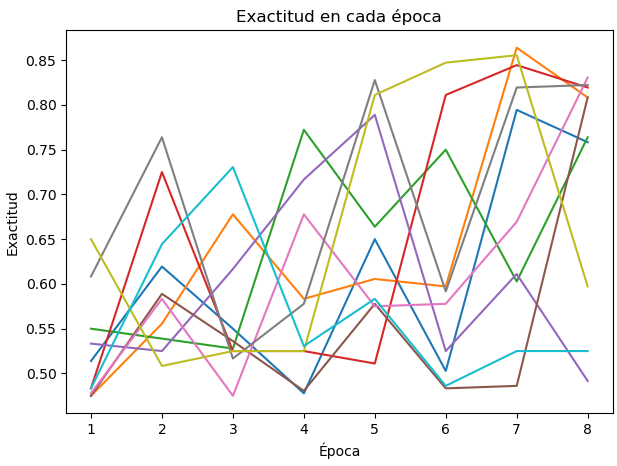
Exactitud para cada subconjunto de datos con la función de activación ‘softmax’

|  |  |  |  |
| --- | --- | --- | --- |
| **Partición** | **Precisión** | | |
| ***Entrenamiento*** | ***Validación*** | ***Testeo*** |
| 1 | 0,672222222 | 0,483333327 | 0,56 |
| 2 | 0,745138889 | 0,563888884 | 0,63 |
| 3 | 0,755555556 | 0,744444443 | 0,79 |
| 4 | 0,697222222 | 0,50555556 | 0,61 |
| 5 | 0,715972222 | 0,808333335 | 0,855 |
| 6 | 0,722916667 | 0.59999999 | 0,69 |
| 7 | 0,670833333 | 0,844444443 | 0,86 |
| 8 | 0,679166667 | 0,841666669 | 0,88 |
| 9 | 0,725694444 | 0,833333328 | 0,865 |
| 10 | 0,725694444 | 0,861111118 | 0,89 |

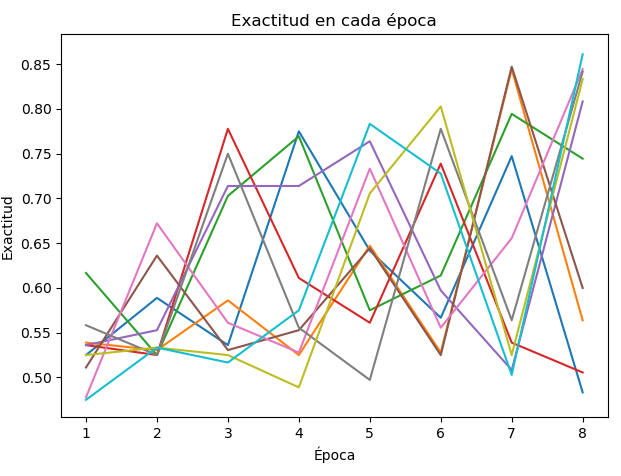
Estadísticas Del la Exactiud para la red convolucional modificada

|  |  |  |  |
| --- | --- | --- | --- |
| **Estadísticas** | ***Entrenamiento*** | ***Validación*** | ***Testeo*** |
| **Promedio** | 0,711041667 | 0,720679012 | 0,763 |
| **Desviación** | 0,008069493 | 0,197810361 | 0,14646 |

Se puede ver entonces que este cambio incrementa el promedio de la exactitud y lo hace más preciso. Esto se puede evidenciar gráficamente en (7 y 8) ya que para la segunda las líneas llegan a puntos más cerca y con valores superiores en la última época



Exactitud en cada época para sigmoid



Exactitud en cada época para sofmax

Sin embargo, la desviación sigue siendo alta y por lo tanto se hacen las modificaciones descritas anteriormente. Logrando que porcentajes superiores de exactitud y reduciendo la desviación aproximada en una cuarta parte, es decir mejorando radicalmente la precisión como se muestra en las siguientes tablas:

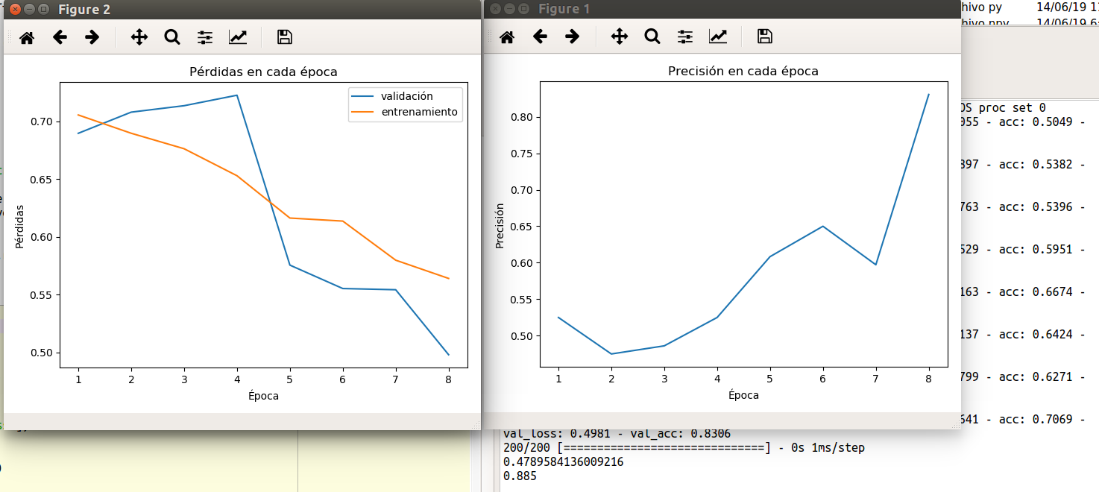
Exactitud para cada subconjunto de datos con la red neuronal final

|  |  |  |  |
| --- | --- | --- | --- |
| **Partición** | **Exactitud** | | |
| ***Entrenamiento*** | ***Validación*** | ***Prueba*** |
| 1 | 0,73125 | 0,75357145 | 0,8283333 |
| 2 | 0,7732143 | 0,60714287 | 0,70666665 |
| 3 | 0,77410716 | 0,86071426 | 0,86833334 |
| 4 | 0,73660713 | 0,76785713 | 0,83166665 |
| 5 | 0,74642855 | 0,80714285 | 0,7866667 |
| 6 | 0,78482145 | 0,7607143 | 0,84 |
| 7 | 0,71339285 | 0,8464286 | 0,865 |
| 8 | 0,73303574 | 0,7821429 | 0,85 |
| 9 | 0,73482144 | 0,85 | 0,85833335 |
| 10 | 0,7580357 | 0,85 | 0,85833335 |

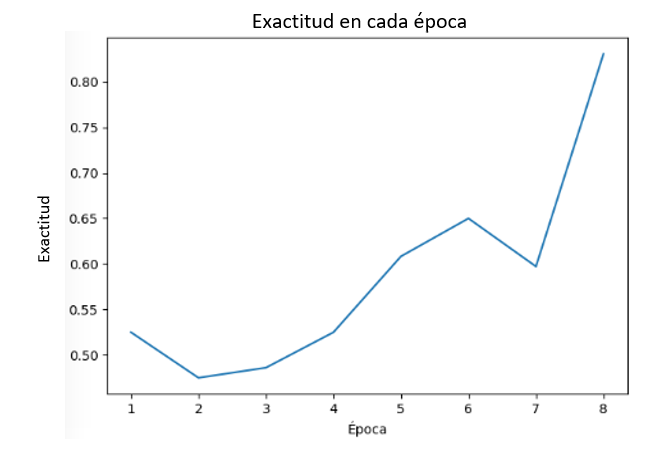
Estadísticas Del la Exactiud para la red convolucional final

|  |  |  |  |
| --- | --- | --- | --- |
| **Estadísticas** | ***Entrenamiento*** | ***Validación*** | ***Prueba*** |
| **Promedio** | 0,748571432 | 0,788571436 | 0,829333334 |
| **Desviación** | 0,004778702 | 0,051831625 | 0,021890004 |

Además, otro logro representativo con respecto a las anteriores redes es que se logran curvas más suaves (8 y 9), lo que permitiría tener una predicción comportamiento del modelo a través de las épocas



Pérdidas en cada época para red neuronal final



Exactitud en cada época para red neuronal final

Análisis de Resultados

Al observar los datos mostrados en las tablas II, IV vs VI se evidencia que no hay comportamiento de trade-off entre la desviación y precisión, ya que ambos aumentan sin importar que este cambo se mucho más significativo en uno que en otro. Cabe resaltar que es una pequeña variación del fenómeno esta discutido en la literatura como el trade-off entre *bias y varianza.* Ya que el que haya errores de un orden de magnitud implica que no logra distinguir del todo las características más relevantes para poder diferenciar de manera acertada, es decir tiene un alto bias o poca exactitud. Mientras que la varianza está ligada a la desviación, e implica que tiene un modelo muy complejo tratando de ajustar un pequeño número de características; se dice entonces que el modelo tiene un sobre ajuste a los datos de entrenamiento y se aprende los datos de memoria y pierde la capacidad de generalizar el modelo para nuevos datos; es decir, entre más parecidos sean los datos a los de entrenamiento mejor desempeño tendrá. Por lo tanto, para lograr mejores soluciones los factores de reducción de dimensión afectivamente ayudaron a disminuir la complejidad del modelo generado. Sin embargo, se configuraron los parámetros adecuados para evitar pasar al punto en el que deja de reconocer las características importantes.

Lo anterior se debe también a la escogencia de los datos de entrenamiento, por lo que se podría decir para los casos en que el modelo era muy simplificado, o en otras palabras había alto bias, era debido a que la escogencia aleatoria los datos casualmente eran similares entre si. Por lo tanto, se podría aumentar la cantidad de datos de entrenamiento si es posible con datos nuevos y si no agregar un generador de imágenes que agregue las mismas imágenes modificadas y encontrar los filtros que mejor extraigan las características buscadas.

También, se observó que la mayoría de las veces el porcentaje obtenido en entrenamiento era menor al de validación y prueba, lo cual podría indicar que talvez se necesitaba un modelo más complejo ya que como se entrenaba con muchos datos este no alcanzaba a poder predecir todos los datos de entrenamiento pero si justo un subset de estos que son similares a los que logra predecir y por tanto se obtenían mejores resultados

Debido a que los resultados se ven influenciados por los datos. Se realiza una última prueba usando random Forest y como criterio la exactitud. Sin embargo, a pesar de que se reduce la desviación a 0,00599 también se reduce el promedio al 0,506. Esto se debe a que no hubo un proceso de caracterización, sino que se convierte la imagen directamente en una matriz, lo cual como se puede ver no es muy recomendable, aunque para algunos casos como para el de [9] podría funcionar mejor.

Conclusiones

Se pudo evidenciar la utilidad del uso de una red convolucional para un problema de astrofísica, además, cabe resaltar que cada vez son más los campos en los que se requiere del procesamiento de imágenes y que cada vez más lo datos que se tienen. Para esto se puede usar machine learning, sin embargo, es necesario identificar el problema si es de clasificación, identificación de objetos dentro de una imagen o clusterización, para utilizar la herramienta requerida.

En cuanto a los parámetros e hiperparámetros, se mostró importante conocer el significado en el modelo de cada uno y su campo de acción con el fin de obtener un mejor modelo. Como por ejemplo el número de filtros, su tamaño, el número de neuronas, entre otros. También es necesario realizar ciertas visualizaciones de los datos ya que estas dan idea de ciertos parámetros como el número de épocas para evitar sobre ajuste en caso de que en la gráfica de la figura 7 las curvas fueran naranja validación y azul entrenamiento, por ejemplo.

Por otro lado, se evidencio la importancia de las bases de datos y la procuración a tener una base de datos amplia y además escoger con mayor robustez los datos de entramiento y prueba. Además, también es necesario tener en cuenta la capacidad computacional, ya que si esta de base de datos o el modelo son muy grandes, debería ser corrido en una GPU.

Finalmente, las posibles mejoras a futuro consistirían en realizar una búsqueda de filtros que mejoren la extracción de características permitiendo así un menor bias y al implementar tanto una partición más robusta de los datos y una base datos más completa, se reduciría la varianza y se suavizarían las curvas de la exactitud por época. O en caso de que estos no funcionen como deberían una nueva búsqueda de parámetros e hiper parámetros pero desde una GPU para agilizar el proceso.

Por otro lado, como trabajo adicional y de contraste realizar el clasificador con el algoritmo de random forest extrayendo las características por medio de un PCA para comparar con los resultados obtenidos por la red convolucional mejorada.

Referencias

SalesForce. 2018. “Machine Learning y Deep Learning: aprende las diferencias”. Disponible en: <https://www.salesforce.com/mx/blog/2018/7/Machine-Learning-y-Deep-Learning-aprende-las-diferencias.html>

E. Eiroa. “Estudios sobre lentes gravitacionales con objetos relativistas”. Universidad de Buenos Aires, Facultad de Ciencias Exactas y Naturales Departamento. Disponible en <https://digital.bl.fcen.uba.ar/download/tesis/tesis_n3448_Eiroa.pdf>

T. Aguirre. “Lentes gravitacionales fuertes: análisis y detección” Disponible en <https://rdu.unc.edu.ar/bitstream/handle/11086/6247/trabajoTagliaferro.pdf?sequence=1&isAllowed=y>

Instituo de Astrofísica de las canarias. “Lentes Gravitacionales”. Disponible en <https://www.iac.es/cosmoeduca/gravedad/fisica/fisica1.htm>

SARKAR D, BALI R., SHARMA T. 2018 Practical Machine Learning

with Python Apress, India.

R. Martinez y P. Protopapas. “Introduction to convolutional networks”. Harvard-Smithsonian Cener for Astrophisics

A. Saxena. 2016. “Convolutional Neural Networks (CNNs): An Illustrated Explanation”. Disponible en\_<https://blog.xrds.acm.org/2016/06/convolutional-neural-networks-cnns-illustrated-explanation/>

A. Rosebrock. “Keras Conv2D y capas convolucionales”. 2018. Disponible en línea en: <https://www.pyimagesearch.com/2018/12/31/keras-conv2d-and-convolutional-layers/>

Hyperion Development. 2019. “Python machine learning: Introduction to image classification”. Disponible en <https://blog.hyperiondev.com/index.php/2019/02/18/machine-learning/>

J. Dean. “Large-Scale Deep Learning for Intelligent Computer Systems”. Disponible en <http://www.wsdm-conference.org/2016/slides/WSDM2016-Jeff-Dean.pdf>

Y. LeCun, Y. Bengio y G. Hinton. . 2015 . “Deep Learning”. Disponible online en: <https://doi.org/10.1038/nature14539>

 Stanford CS class. Sf. “Convolutional Neural Networks (CNNs / ConvNets)“ Disponible en linea en http://cs231n.github.io/convolutional-networks/#overview