Universidad Nacional Autónoma de México

Facultad de Ciencias

Redes Neuronales Artificiales aplicadas a la Cosmología Observacional

T E S I S

que para obtener el título de:

licenciado en física

P R E S E N T A :

Juan de Dios Rojas Olvera

tutor

Dr. José Alberto Vázquez González

Facultad de Ciencias. UNAM, 2022

CONSTANCIA DE PRESENTACIÓN DE EXAMEN PROFESIONAL

En la Universidad Nacional Autónoma de México, en el AULA SOTERO PRIETO I de la FACULTAD DE CIENCIAS siendo las 12:00 horas del día 13 de enero de 2023, el alumno:

JUAN DE DIOS ROJAS OLVERA

de nacionalidad MEXICANA con número de cuenta 313307579 se presentó con el fin de sustentar el examen oral para obtener el título de:

FÍSICO

en su modalidad de titulación por TESIS con el trabajo titulado: Redes Neuronales artificiales aplicadas a la Cosmología Observacional. El alumno cursó sus estudios en el periodo 2016-1 a 2022-2 habiendo obtenido un promedio de 8.55 y cumpliendo con los requisitos académicos señalados en el plan de estudios 1081 aprobado por el H. Consejo Universitario.

El Jurado designado por el Comité Académico integrado por:

PRESIDENTE: DR. JOSE ANTONIO DE DIEGO ONSURBE
VOCAL: DR. J. GUADALUPE PEREZ RAMIREZ
SECRETARIO: DR. JOSE ALBERTO VAZQUEZ GONZALEZ
SUPLENTE: DR. ANTONIO CASTELLANOS RAMIREZ
SUPLENTE: DR. SEBASTIEN MICKAEL MARC FROMENTEAU

Tras el interrogatorio y deliberación resolvió otorgarle la calificación de:

Aprobado

Procediendo a informarle el resultado, tomarle la Protesta Universitaria y dar por concluido el acto.

PRESIDENTE DEL JURADO | **SECRETARIO DEL JURADO** | **VOCAL DEL JURADO**

El Titular de la entidad académica hace constar que las firmas electrónicas que anteceden son válidas y corresponden a los miembros del jurado.

"POR MI RAZA HABLARÁ EL ESPÍRITU"
DR. VÍCTOR MANUEL VELÁZQUEZ AGUILAR
DIRECTOR DE LA ENTIDAD ACADÉMICA

QR de verificación

Cadena de verificación digital

fdbb3ba000a7d0f49552c038df624f03fd7654ce8095c5a0c1ec0bbd0ed893b38f7f004e6fd546325152976b07528c8efff0c877fb89db93ef56e75bb260dfe8

FACULTAD DE CIENCIAS

FACULTAD DE CIENCIAS
DEPARTAMENTO DE FÍSICA
COORDINACIÓN DE LA LICENCIATURA DE FÍSICA
Asunto: Registro de Director y Tema de tesis

Dr. José Alberto Vázquez González

Instituto de Ciencias Físicas

Universidad Nacional Autónoma de México

P r e s e n t e:

Me permito comunicar a usted, que el Comité Académico de la Licenciatura de Física de esta Facultas ha dado su aprobación para que el alumno, "Juan de Dios Rojas Olvera", con número de cuenta "313307579", de la Licenciatura de Física realice el trabajo de Tesis bajo su dirección, con el título: "Redes Neuronales aplicadas a la cosmología observacional y estadística de energía oscura".

Sin más por el momento, me despido enviándole un cordial saludo.

A T E N TA M E N TE
"POR MI RAZA HABLARÁ EL ESPÍRITU"
Cd. Universitaria, CD.MX., a 8 septiembre del 2021
COORDINADOR DE LA LICENCIATURA EN FÍSICA

FACULTAD DE CIENCIAS

DR. GERARDO JORGE VÁZQUEZ FONSECA

DEPARTAMENTO DE FÍSICA
LIC. FÍSICA

A todas las personas que han sacrificado mucho
por aportar un poco a la ciencia...

Agradecimientos

- Agradezco al Dr. José Alberto Vázquez por ser mi guía en la recta final de la carrera, por la atención y el tiempo dedicados en el trabajo que realizamos juntos, por haberme dado la oportunidad de formar parte de su equipo de trabajo y finalmente, por ser un gran investigador y mejor ser humano.

- Quiero agradecer a mis sinodales: el Dr. Antonio Castellanos, el Dr. Sebastien Fromenteau y el Dr. José Antonio De Diego. Por dedicar tiempo a la lectura de esta tesis, por sus observaciones y consejos, los cuales agregaron gran valor a la realización de este trabajo.

- Agradezco a mis profesores, quienes me transmitieron parte de sus conocimientos, experiencia y pasión por la ciencia. Ellos han sido un faro de luz en este camino.

- Quiero hacer una mención especial al Dr. Isidro Gómez Várgas, quien formó parte importante de este trabajo con sus consejos y su experiencia.

- A mis padres María de los Ángeles Olvera y Juan Rojas, retribuirles un poco de lo que me han dado. Por su apoyo, esfuerzo y enseñanzas, sin los cuales nada de esto sería posible.

- Agradezco mis tíos Evangelina Olvera y Félix Ocampo, así como a mi primo Félix. Por ser para mi como una segunda familia y por todo el apoyo que me han brindado en los momentos difíciles.

- Agradezco por el apoyo brindado a FOSEC SEP-CONACYT Investigacion Básica A1-S-21925, Ciencias de Frontera CONACYT-PRONACES/304001/202, y

UNAM-DGAPA-PAPIIT IA104221. Pues sus contribuciones económicas fueron de gran ayuda para la realización de este trabajo.

- Agradezco al Instituto de Ciencias Físicas por alojarme como su alumno y permitirme trabajar en tan digna institución. De igual manera, a mi *alma máter*: La Facultad de Ciencias de la UNAM, pues le debo mi formación como Físico.

- Finalmente agradezco a mis compañeros de generación, quienes hicieron que mi paso por la facultad fuera más disfrutable. A todos ellos, gracias!

Juan de Dios Rojas Olvera

Even when a light fills a space
Even though the space is a hole
It all began with no one...

Even though the night hurts so much
Even though the mourning is so low
It all began with no one...

Hey when the light spells a name
All the people who were the ones you loved
They wait for you;

Said and done
All the people before,
As they reach for the harbour of a thought
They wait for us.

— Kate Louise Smith, Aurosonic & Denis Karpinskiy

Resumen

El tratamiento de los datos se ha vuelto una parte fundamental del proceso científico, especialmente en cosmología. Desde el análisis de datos observacionales que ponen a prueba los modelos teóricos del Universo, hasta las simulaciones a gran escala del cosmos. El objetivo principal de este trabajo es entonces, proponer a las redes neuronales artificiales como una herramienta versátil y poderosa, capaz de simplificar y optimizar distintos procesos comunes en la física computacional, así como llevar a cabo tareas importantes en la astronomía y la cosmología.

Aquí se presenta una revisión a los fundamentos matemáticos y técnicos del aprendizaje profundo. También se mostrará su desempeño en el modelado de datos cosmológicos, tareas numéricas (ahorro de tiempo computacional) y la clasificación de objetos estelares. El contenido de este trabajo también puede ser consultado en su forma compacta en el artículo '*Observational Cosmology with Artificial Neural Networks*' [1].

Abstract

Data processing has become a fundamental part of the scientific process, especially in cosmology. From the analysis of observational data that test theoretical models of the Universe, to large-scale simulations of the cosmos. Then, the main aim of this work is to propose artificial neural networks as a versatile and powerful tool, capable of simplifying and optimizing different common processes in computational physics, as well as carrying out important tasks in astronomy and cosmology.

A review of the mathematical and technical foundations of deep learning is presented here. Its performance in cosmological data modeling, numerical tasks (saving computational time) and stellar object classification will also be shown. The content of this work can also be consulted in its compact form in the paper '*Observational Cosmology with Artificial Neural Networks*' [1].

Índice general

Índice de figuras

1 | Introducción

A lo largo de las últimas décadas, la sinergia entre ciencia y tecnología se ha visto reforzada. La matemática y la física son el pilar de las innovaciones tecnológicas surgidas en el siglo XX y XXI; por otro lado, sería muy difícil imaginar hacer ciencia en la actualidad sin la ayuda de una computadora o cualquiera de las herramientas digitales existentes. Desde los microscopios electrónicos que posibilitan visualizar el interior de las células, hasta los telescopios que han permitido ver más allá de la vía láctea y cuyos datos inherentemente deben ser procesados de forma computacional.

Sería plausible considerar que al paradigma científico de hipótesis-experimentación herencia del renacimiento, se le podría agregar un tercer elemento: la simulación por ordenador. Más aún, un tratamiento intensivo de los datos es potencialmente el cuarto elemento. El avance tecnológico ha repercutido en la física permitiendo el acceso a herramientas de medición cada vez más numerosas y sofisticadas. En el caso de la cosmología observacional y la astronomía esto es especialmente cierto. Es de resaltar la tasa obtenida al comparar el volumen de datos procedentes de satélites, telescopios, instrumentos de alto rendimiento, redes de sensores, aceleradores y superordenadores en la década de los 90 con los correspondientes al 2000, se obtiene que estos han aumentado en una proporción de entre 10^2 y 10^3 [2].

Una muestra de esta masificación de los datos, se encuentra en uno de los eventos más importantes para la ciencia en los tiempos recientes: la obtención de la primer imagen de un agujero negro. Este evento no sólo confirmó la existencia de estos objetos tan exóticos, sino que reforzó las bases observacionales de la relatividad y por extensión, de la cosmología. Obtener la imagen del agujero negro en el centro de la galaxia Messier 87 fue una labor ardua desde el punto de vista computacional y

(según se menciona en [3]) supuso un desafío para las herramientas de procesamiento de datos existentes. El volumen de datos recolectados durante las observaciones están en el rango de los petabytes (10^{15} bytes). El tiempo de cálculo fue otro de los retos a superar, debido a la enorme cantidad de parámetros diferentes, además el trabajo de integrar todas las observaciones de las distintas estaciones (lugares alrededor del planeta donde se encontraban los telescopios) [3]. Y todo lo anterior sin siquiera entrar en el proceso de generación de la imagen, tarea de reconstrucción imposible de hacer sin métodos computacionales de primer nivel.

Esto representa el presente en la astronomía, y el manejo de datos tradicional no basta para navegar en estas aguas. Esta tendencia tenderá a continuar de la misma manera en los próximos años. En 2022 comenzarán las operaciones del observatorio Vera C. Rubin en Chile: se estima que podrá recibir hasta 20 Tbytes de información por noche, aproximadamente la cantidad recolectada por el Sloan Digital Sky Survey entre 2000 y 2010, proyecto gracias al cual se han conseguido los mapas 3D del universo más detallados hasta el momento. Así mismo, los telescopios Square Kilometre Array en Australia-Sudáfrica, para el que se estima recibir hasta 2 petabytes diarios después de su inauguración en 2028 y el next-generation Very Large Array (ngVLA) en Nuevo México, generará cientos de petabytes anualmente, algunas cientos de veces más que el VLG actual [4]. Finalmente, mientras estas líneas son escritas, el telescopio espacial James Web, se encuentra en camino a tomar la que será su orbita alrededor de la Tierra, el telescopio más poderoso construido hasta la fecha y que se espera traiga consigo información nunca antes vista por los astrónomos. Por todo lo anterior, es vital adoptar técnicas que ayuden a los astrónomos y cosmólogos a sacar el mayor provecho a esa avalancha de datos.

Por otro lado, realizar simulaciones de fenómenos físicos es una tarea complicada y costosa desde el punto de vista computacional, pero que puede resultar muy útil para comprender mejor lo que se está estudiando o como una base para realizar predicciones. En cosmología este tópico es muy importante, pues hay muchas situaciones que nunca podrán ser observadas y la única manera de interactuar con ellas es a través de una simulación. Además, estas simulaciones podrían preceder a los datos

observacionales del fenómeno.

Por ejemplo, en 2013 el físico Kip Thorne (Nobel de Física 2017) colaboró con la empresa Double Negative Gravitational Renderer (DNGR) con el objetivo de por primera vez llevar a la pantalla grande la visión realista que tendría una persona al estar cerca de un agujero negro. Thorne desarrolló un código que permitió representar la propagación de un haz de luz a través del espacio-tiempo curvado. Fue así como se obtuvo la famosa simulación de *gargantúa*, el agujero negro giratorio de la película *Interestellar*. Esta colaboración entre conocimiento físico y capacidad de despliegue que a veces sólo una empresa cinematográfica posee, dio como resultado una simulación capaz de explorar los efectos de lente gravitacional en un agujero negro con una calidad y dinámica de imagen sin precedentes (23 millones de píxeles por imagen) [5]. Esta predicción sobre el aspecto de un agujero negro y su disco de acreción luego sería reforzada por la imagen del agujero negro en el centro de Messier 87.

Para la cosmología también es muy importante la comprensión de la estructura y la formación de galaxias en el Universo. Es por ello que hacer simulaciones sobre este proceso es fundamental para avanzar en su estudio, aunque esta no es tarea fácil, pues la cantidad de procesos físicos que intervienen aquí es cuantiosa, además de que depende directamente de los métodos numéricos y la capacidad de cómputo. Por ejemplo, es posible realizar simulaciones sobre la materia oscura, la energía oscura y la materia ordinaria en un espacio-tiempo en expansión. O bien a partir de algunas condiciones iniciales bien definidas, modelar la formación y evolución del universo, utilizando el método de $N-$cuerpos (para un ejemplo, revisar [6]). También se han realizado simulaciones hidrodinámicas de la formación de galaxias, las cuales han permitido hacer predicciones precisas sobre la agrupación de materia a escala cosmológica [7].

Con todo lo anterior, este trabajo tiene la intención de implementar una herramienta de vanguardia perteneciente a la inteligencia artificial, conocida como *Redes Neuronales Artificiales* (RNA). Se trata de un paradigma de la inteligencia artificial que ha demostrado rendir buenos resultados tanto para el manejo de datos como en simulaciones en física.

Algunos ejemplos de aplicaciones de las RNA en este campo del conocimiento se encuentran en la física de altas energías, donde los datos generados por los aceleradores de partículas pueden ser bastante complejos y de alta dimensionalidad. Por otro lado, las redes neuronales han ayudado a acelerar el tiempo de cómputo en cosmología [8, 9]. Por ejemplo, en [10] a los datos procedentes de la radiación del Fondo Cósmico de Microondas (CMBr) se implementaron varios tipos de RNA en la estimación de parámetros, que aceleraron el proceso de inferencia en un factor superior a $O(4)$ [10]. Además, las RNAs se han utilizado para realizar reconstrucciones no paramétricas de funciones cosmológicas [11, 12, 13]. Además, la implementación de redes neuronales ha permitido separar las señales de distinto origen en las recolecciones de datos, o la elaboración de mapas para el CMBr libres de contribuciones externas. En [14] se implementó una red neuronal que pudo separar eficientemente señales que incluían: emisiones de polvo térmico, CMBr, sincrotrón galáctico y radiación emitida por cúmulos de galaxias. Del mismo modo, en [15], los autores entrenaron una red para reconocer el ruido en los datos y separarlo de las fluctuaciones en la temperatura del CMBr. El Deep Learning también se ha utilizado en astrofísica observacional para la detección y clasificación de cuásares en las observaciones realizadas por el Sloan Digital Sky Survey (SDSS). La red neuronal de este trabajo fue capaz de ofrecer 175 nuevos candidatos a cuásar [16]. Se han implementado redes neuronales artificiales para resolver ecuaciones diferenciales [17] y para resolver integrales numéricamente de forma más eficiente [18].

Dentro de todas las aplicaciones se encuentra la presencia de distintos tipos de redes neuronales, cada una con sus particularidades. Por ejemplo: Redes Neuronales Convolucionales, pieza importante dentro de la visión por ordenador y el análisis de imágenes o vídeos; Redes Neuronales Recurrentes, conocidas por su capacidad para procesar y obtener información de datos secuenciales como en el procesamiento de lenguaje; o las Redes Neuronales Bayesianas, comúnmente utilizadas para procesos de inferencia estadística de parámetros. Aquí se cubrirá únicamente el tipo de red conocido como perceptrón multicapa o de tipo *capas totalmente conectadas.* Estas representan el modelo fundamental de las redes neuronales, y su comprensión es clave

para la implementación de los otros tipos existentes mencionados anteriormente.

Una vez detallado el contexto actual de la Cosmología Observacional, la hipótesis de la que parte este trabajo es que las redes neuronales artificiales son una herramienta poderosa y versátil que conviene adoptar dentro del paradigma cosmológico. También, el uso de las técnicas propias del aprendizaje profundo ofrecerán una manera muy buena de describir funciones subyacentes en conjuntos de datos, reducir el tiempo de cómputo de muchas tareas numéricas y facilitar algunas actividades propias de la astronomía, como la clasificación de objetos astronómicos.

Este trabajo comenzará con el Capítulo 2, un breviario de los conceptos más importantes concernientes al desarrollo de la cosmología así como de las ecuaciones esenciales que dan formalidad a esta área y que serán utilizadas en el resto de capítulos. Luego se presentarán los fundamentos del aprendizaje profundo mediante un repaso de conceptos y formalismo matemático en los Capítulos 3, 4 y 5. En el Capítulo 6 se desarrollarán y presentarán los problemas de investigación que forman el cuerpo de este trabajo y que buscan probar la hipótesis planteada. Finalmente los comentarios finales, reflexiones y conclusiones serán expuestos en el Capítulo 7.

2 | El universo y sus ecuaciones

'El cosmos es todo lo que es, todo lo que fue y todo lo que será...'
Estas son las palabras con las que Carl Sagan comienza su libro más famoso: *Cosmos*. A pesar de su vastedad, la humanidad siempre se ha empeñado en comprender al universo y su lugar en él: ya sea a través de la religión, la filosofía o la ciencia. Es la cosmología la rama de la ciencia encargada de estudiar al universo, su contenido, su evolución y en última instancia teorizar sobre su origen y final.

En física, es imprescindible que las teorías formuladas a partir de fenómenos naturales cuenten con una base matemática sólida. Este formalismo aporta rigor a la ciencia y permite, en última instancia, hacer predicciones y descripciones acertadas del fenómeno. Una de las piedras angulares de la cosmología es la *Teoría de la Relatividad*, formulada por Albert Einstein y constituida por dos partes esenciales: las teorías de relatividad especial y general, publicadas en 1905 y 1915 respectivamente. En relatividad especial se estudia el movimiento de los cuerpos en sistemas de referencia inerciales, es decir, aquellos que describen un estado de movimiento uniforme entre sí, tales que no es posible distinguir unos de otros mediante experimentos puramente mecánicos. Aquí se concibieron cambios importantes en la física, que en su momento rompieron con algunos de los paradigmas clásicos muy arraigados hasta ese momento. Por ejemplo, la velocidad de la luz como límite universal del movimiento para objetos con masa, la equivalencia entre masa y energía o la relatividad del tiempo [19].

Para Newton, los objetos podían ser clasificados según su naturaleza. A saber, estos podían ser: absolutos o relativos, verdaderos o aparentes, matemáticos, etc. Según él mismo describió en las primeras definiciones con las que comienza su obra *'Philosophiæ Naturalis Principia Mathematica'*, el tiempo debe entenderse en los siguientes

términos:

> *'El tiempo absoluto, verdadero y matemático, por sí mismo y por su propia naturaleza fluye equitativamente sin tener en cuenta nada externo...'*

Caso similar para el espacio:

> *'El espacio absoluto, en su propia naturaleza, sin tener en cuenta nada externo, permanece siempre similar e inmóvil...'*

Sin embargo, Einstein concibió al espacio y el tiempo como dos manifestaciones de un mismo ente más abstracto y general: *el espacio-tiempo*. Más aún, el espacio-tiempo era relativo, idea contraria a la de Newton, donde espacio y tiempo se concebían independientes y absolutos. También encontró un nuevo entendimiento para la gravedad dentro de su teoría, el cual puede resumirse en la famosa frase de John Wheeler: *La materia le dice al espacio-tiempo cómo curvarse, el espacio-tiempo curvado le dice a la materia cómo moverse.* La teoría general de la relatividad hace una descripción geométrica de la gravedad a través de las ecuaciones de campo de Einstein [20, pp 29]:

$$G^{\mu\nu} + \Lambda g^{\mu\nu} = -\frac{8\pi G}{c^4} T^{\mu\nu}, \tag{2.1}$$

donde $G^{\mu\nu}$ es conocido como el tensor de Einstein, que ofrece una descripción del espacio-tiempo; $T^{\mu\nu}$ como el tensor de energía-momento; $g^{\mu\nu}$ es el tensor asociado a la métrica del espacio-tiempo; Λ es una constante añadida por Einstein, denominada comúnmente *constante cosmológica*; finalmente, G y c son las constantes de gravitación y velocidad de la luz respectivamente. Estas ecuaciones son en cierta forma uno de los pilares de la cosmología y son esenciales para el estudio de infinidad de fenómenos.

Respecto a la distribución del contenido del espacio-tiempo, suelen hacerse algunas fuertes suposiciones que facilitan en gran medida las cosas, por ejemplo: **A grandes escalas (100 Mpc o más), la distribución de dicho contenido es homogénea e isotrópica**. La isotropía significa que no existe una dirección privilegiada en el cosmos: se mire en la dirección en que se mire, todo lucirá igual. La homogeneidad

indica que cualquier punto en el universo será idéntico a cualquier otro: los cosmólogos suelen llamar a esta suposición como el *principio Copernicano*, y dice que *'no hay nada especial acerca de nuestra localización en el cosmos'.* El principio cosmológico es entonces, asumir que el universo es homogéneo e isotrópico a gran escala. Esto facilita una descripción geométrica del espacio-tiempo, pues de esta manera su geometría, definida por la curvatura, deberá ser constante en cualquier punto donde se mida.

En la década de los 20, Friedmann (1922) y Lemaître (1927) encontraron soluciones a las ecuaciones de campo de Einstein. Aunque a sus resultados no se les dio demasiada importancia en primera instancia, en las soluciones propuestas por Lemaître se mostraba que todas las galaxias debían estarse alejando unas de otras, debido a un universo en expansión, algo que años más tarde sería comprobado por las observaciones de Hubble. Finalmente, en 1929 basadas en el trabajo de Robertson y Walker, salieron a la luz las soluciones homogéneas e isotrópicas para las ecuaciones de Einstein, que describían un universo en expansión [21]. Todo esto concluye con la métrica que describe ese espacio-tiempo y que hoy lleva sus iniciales: *La métrica Friedmann-Lemaître-Robertson-Walker* $(FLRW)$. Esta métrica define el intervalo infinitesimal de distancia en el espacio-tiempo para coordenadas esféricas (r, θ, ϕ):

$$ds^2 = dt^2 - a^2(t)\left[\frac{dr^2}{1-kr^2} + r^2(d\theta^2 + sin^2\theta\, d\phi^2)\right]. \tag{2.2}$$

Donde r es la coordenada radial comóvil; $a(t)$ es conocido como el factor de escala adimensional, cuyo valor medido al día de hoy es $a_0 = 1$; k es una constante que representa la geometría del espacio-tiempo, denominada constante de curvatura. Con esto, se cuenta con el tensor métrico:

$$g^{\mu\nu} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{-a(t)^2}{1-kr^2} & 0 & 0 \\ 0 & 0 & -a(t)^2 r^2 & 0 \\ 0 & 0 & 0 & -a(t)^2 r^2 sin^2\theta \end{pmatrix}. \tag{2.3}$$

A partir de él es posible encontrar el tensor de Einstein. Que está definido como:

$$G_{\mu\nu} = R_{\mu\nu} - \frac{1}{2} g_{\mu\nu} R. \tag{2.4}$$

Donde $R_{\mu\nu}$ y R son el tensor y escalar de Ricci respectivamente. A su vez, ambas expresiones se pueden calcular a partir del tensor de Riemann $R^{\alpha}_{\mu\beta\nu}$ usando:

$$R_{\mu\nu} = R^{\beta}_{\mu\beta\nu},$$

$$R = R^{\mu\nu} g_{\mu\nu}.$$

A su vez, el tensor de Riemann está definido por los símbolos de Christoffel del espacio-tiempo:

$$R^{\alpha}_{\mu\beta\nu} = \partial_\beta \Gamma^{\alpha}_{\mu\nu} - \partial_\nu \Gamma^{\alpha}_{\mu\beta} + \Gamma^{\alpha}_{\beta\lambda} \Gamma^{\lambda}_{\mu\nu} - \Gamma^{\alpha}_{\nu\lambda} \Gamma^{\lambda}_{\mu\beta}. \tag{2.5}$$

El tensor métrico 2.3 es diagonal y esta simetría hará más sencilla la obtención de los símbolos de Christoffel, definidos como:

$$\Gamma^{\alpha}_{\mu\nu} = \frac{1}{2} g^{\alpha\beta} \left[\partial_\mu g_{\beta\nu} + \partial_\nu g_{\beta\mu} - \partial_\beta g_{\mu\nu} \right]. \tag{2.6}$$

Por lo que, los símbolos de Christoffel que no se anulan son (recordar que los índices inferiores μ, ν conmutan):

- $\Gamma^{t}_{r,r} = \frac{a\dot{a}}{1-k^2r^2}$,
- $\Gamma^{t}_{\theta,\theta} = r^2 a\dot{a}$,
- $\Gamma^{t}_{\phi,\phi} = r^2 a\dot{a} \sin^2\theta$,
- $\Gamma^{r}_{t,r} = \Gamma^{\theta}_{t,\theta} = \Gamma^{\phi}_{t,\phi} = \frac{\dot{a}}{a}$,
- $\Gamma^{r}_{r,r} = \frac{kr}{(1-kr^2)}$,
- $\Gamma^{r}_{\theta,\theta} = -r(1-kr^2)$,
- $\Gamma^{r}_{\phi,\phi} = -r(1-kr^2)\sin^2\theta$,

- $\Gamma^{\theta}_{r,\theta} = \Gamma^{\phi}_{r,\phi} = \frac{1}{r}$,
- $\Gamma^{\theta}_{\phi,\phi} = -\sin\theta\cos\theta$,
- $\Gamma^{\phi}_{\phi,\theta} = \cot\theta$.

Así, al calcular las componentes del tensor de Ricci contrayendo el tensor de Riemann, se tiene que los únicos distintos de cero son aquellos en la diagonal:

- $R_{tt} = R^{\lambda}_{t\lambda t} = -3\frac{\ddot{a}}{a}$,
- $R_{rr} = R^{\lambda}_{r\lambda r} = \frac{a\ddot{a}}{1-kr^2} + \frac{2\dot{a}^2}{1-kr^2} + \frac{2k}{1-kr^2}$,
- $R_{\theta\theta} = R^{\lambda}_{\theta\lambda\theta} = r^2 a\ddot{a} + 2r^2\dot{a}^2 + r^2 k$,
- $R_{\phi\phi} = R^{\lambda}_{\phi\lambda\phi} = r^2 a\ddot{a} + 2r^2\dot{a}^2 + 2kr^2$.

Por otro lado, el escalar de Ricci queda como:

$$R = -6\left(\frac{\ddot{a}}{a} + \left(\frac{\dot{a}}{a}\right)^2 + \frac{k}{a^2}\right).$$

Ahora, como consecuencia de asumir al universo isotrópico este será descrito por un tensor de energía-momento que dependerá únicamente de ρ, que representa la densidad total del contenido del cosmos y p su presión. Entonces este tensor también será diagonal cuyas componentes no nulas son:

$$T_{tt} = \rho g_{tt},$$

$$T_{ii} = -p g_{ii}.$$

Una vez deducidos todos estos elementos, ya pueden ser evaluados en las ecuaciones de Einstein. Como todos los tensores involucrados son diagonales, se pueden analizar únicamente por las ecuaciones obtenidas componente a componente. Para la parte temporal se tiene:

$$R_{tt} - \frac{1}{2}R g_{tt} - \Lambda g_{tt} = 8\pi G\rho,$$

que es equivalente a:

$$3\left(\frac{\dot{a}}{a}\right)^2 + \frac{3k}{a^2} - \Lambda = 8\pi G\rho,$$

cuyo desarrollo concluye con la ecuación:

$$\left(\frac{\dot{a}}{a}\right)^2 = \frac{\kappa_0}{3}\rho - \frac{k}{a^2(t)} + \frac{\Lambda}{3}, \tag{2.7}$$

donde $\kappa_0 = 8\pi G$. Aunque quedan tres ecuaciones correspondientes a las componentes espaciales, en las tres se llega a la misma expresión.

$$\frac{-g_{ii}}{a^2(t)}\left(a\ddot{a} + 2\dot{a}^2 + 2k\right) - \frac{1}{2}Rg_{ii} - \Lambda g_{ii} = 8\pi G(-p)g_{ii}.$$

Nuevamente, el desarrollo de esta expresión lleva a la ecuación:

$$\left(\frac{\ddot{a}}{a}\right) = -\frac{\kappa_0}{6}(\rho + 3p) + \frac{\Lambda}{3}. \tag{2.8}$$

En ocasiones, estas expresiones suelen simplificarse considerando a la constante Λ como una densidad del vacío (como se verá un poco más adelante), y es incluida con el total de las densidades dentro del término ρ. Además, el parámetro de Hubble se define en términos del factor de escala $H \equiv \frac{\dot{a}}{a}$, por lo que las ecuaciones 2.7, 2.8 comúnmente se escriben como:

$$H^2 = \frac{\kappa_0}{3}\rho - \frac{k}{a^2(t)}, \tag{2.9}$$

$$\dot{H} + H^2 = \left(\frac{\ddot{a}}{a}\right) = -\frac{\kappa_0}{6}(\rho + 3p). \tag{2.10}$$

De esta manera, Friedmann dedujo este par de ecuaciones. Es por esto que la ecuación 2.9 es conocida como la *Ecuación de Friedmann*, que describe un universo no estático. Si este se expande o contrae dependerá de los parámetros de curvatura, expansión y densidad. La ecuación 2.10 es conocida como la *Ecuación de desaceleración*, que representa el cambio en la velocidad a la que el universo se expande (o contrae) con respecto del tiempo. Se puede concluir de ella que el cambio en la velocidad a la

que se alejan las galaxias, disminuye si la suma de la presión y densidad es positiva; de manera análoga si la suma de estas contribuciones fuese negativa (más adelante se explicará por qué puede pasar eso), el universo tendería a acelerar la expansión. En cuanto al contenido del cosmos, éste suele resumirse en el término de densidad ρ. Las distintas componentes se agrupan sumando sus densidades: $\rho = \sum_i \rho_i$, donde el índice i representa a las posibles contribuciones: radiación (r), materia bariónica y oscura (m) y energía oscura (Λ). Las características esenciales de cada componente son:

- La materia bariónica incluye toda la materia compuesta por bariones, pero en astronomía este término se usa de manera más general para referirse a toda la materia visible, es decir: aquella formada por protones y neutrones. Formalmente no se debería contar a los electrones por ser leptones, sin embargo, este término sí los engloba cuando estos forman parte de los átomos.

- La radiación incluye a las partículas relativistas. Estos son los fotones, la partícula de la radiación electromagnética. Un tipo de bosón de masa cero y que puede interactuar con los bariones y los electrones. En ocasiones también se incluye a los neutrinos dentro del término radiación, un tipo de fermión sin carga que sólo se ve afectado por la interacción gravitacional y la débil.

- La materia oscura es un tipo de materia teorizada por los cosmólogos a partir de algunos problemas reales, por ejemplo a partir de la rotación de algunas galaxias espirales, cuya velocidad de giro a grandes radios es mayor a la esperada [22, pp 63-73]. Se cree que este tipo de materia no interactúa con el resto salvo de manera gravitacional, pues hasta ahora esa es la única evidencia de su existencia. En este trabajo se agruparán la materia oscura y bariónica en un único término: materia m.

Ahora, respecto al parámetro de curvatura k, éste indica el tipo de geometría que toma el espacio-tiempo: Si $k = 0$, se tiene que la curvatura es la de un espacio-tiempo plano; para $k < 0$, se dice que el universo es abierto con geometría hiperbólica; finalmente si $k > 0$, el universo es cerrado y tiene geometría esférica. La curvatura

está relacionada con el contenido del universo y con su parámetro de expansión, así que dada la ecuación de Friedmann 2.9, es posible encontrar una densidad especial para el cuál la curvatura sea cero, este valor es conocido como la *densidad crítica*:

$$\rho_c = \frac{3H^2}{\kappa_0}. \tag{2.11}$$

A partir de la densidad crítica, es posible definir un nuevo parámetro de densidad normalizado y adimensional, definido como:

$$\Omega_i = \frac{\rho_i}{\rho_c} = \frac{\kappa_0}{3H^2}\rho_i, \tag{2.12}$$

Sustituyendo los parámetros de densidad en la ecuación de aceleración 2.10, es posible definir un nuevo parámetro conocido como el *parámetro de desaceleración*, que permite deducir si la expansión del universo está disminuyendo o acelerando.

$$q_0 = \frac{\Omega}{2} + \frac{3}{2}\sum_i \Omega_i p_i, \tag{2.13}$$

donde $\Omega = \sum_i \rho_i/\rho_c$.

Una de las cuestiones más recurrentes en cosmología era el porqué el universo se mantiene en expansión a pesar de los efectos de la materia y la gravedad. Una de las posibles respuestas proviene de la constante introducida por Einstein, donde el espacio parece poseer propiedades tales como su propia densidad de energía Ω_Λ, la cual parece no diluirse con la expansión del universo. Más aún, el parámetro de desaceleración fue estimado por algunos cosmólogos con ayuda de supernovas de tipo Ia, un tipo de supernova que ocurre por la explosión de una enana blanca en un sistema conformado por dos estrellas. Las observaciones mostraron que de hecho el universo se expande cada vez más rápido. Esto favoreció modelos en los que la energía de densidad aportada por un cierto componente de presión negativa debía ser más del doble que todo el aportado por la materia y radiación [23]. En otras palabras: el requisito más general para explicar la expansión acelerada del universo, es que actualmente el cosmos debería ser dominado por un componente con presión negativa.

La propuesta más simple es la de considerar a la constante cosmológica como un parámetro de densidad Ω_Λ. Hoy en día existen modelos donde la energía oscura o constante cosmológica Λ juega un papel crucial, por ejemplo el modelo estándar: ΛCDM, *Lambda-materia oscura fría* (por sus siglas en inglés) es el modelo estándar de la cosmología contemporánea. En este modelo se han estimado los parámetros de densidad Ω_i y el de Hubble, actualmente con un 68 % de confianza, usando los datos del satélite *Planck* tomados del fondo cósmico de microondas [24]. A partir de dichas mediciones se han estimado valores de: $\Omega = 0.99$, $\Omega_{\Lambda,0} = 0.685 \pm 0.007$, $\Omega_m = 0.315 \pm 0.007$ y $h_0 = 0.674 \pm 0.005$, con h la normalización del parámetro de Hubble. De donde se concluye que en la actualidad la energía oscura abarca casi el 70 % del contenido del universo. Respecto al parámetro de curvatura, este depende de los valores del resto de parámetros, la combinación de los valores antes mencionados nos dan indicios que k es aproximada a cero, lo que indica que el universo pareciera tener una geometría plana.

Es necesario mencionar que ΛCDM aún tiene algunos puntos débiles como teoría. Por ejemplo: se sabe poco sobre la naturaleza de la materia y la energía oscura. Por otro lado, aún existen tensiones respecto al valor real que debería tener el parámetro de Hubble hoy en día, pues aunque las técnicas de medición se han refinado cada vez más, los distintos resultados que se han obtenido hasta ahora con base en las observaciones, por un lado algunos como el mencionado valor obtenido a partir del CMB, centrados en $h_0 = 0.67$ y los obtenidos a partir de supernovas (ver capítulo 3.5) y cuyos valores se acumulan alrededor de $h_0 = 0.71$ [25] . Así que las observaciones respecto a este parámetro parecen no reconciliarse en un valor común.

2.1 Dinámica y evolución de un Universo homogéneo

De la combinación de las dos ecuaciones independientes de Friedmann 2.9, 2.10 se puede deducir una relación fundamental en cosmología. Al derivar respecto al tiempo

la ecuación 2.9 y despejando $\dot{H}$ se obtiene:

$$\dot{H} = \frac{1}{2H}\left(\frac{\kappa_0}{3}\dot{\rho} + \frac{2\dot{a}}{a^3}\right).$$

Esta expresión puede insertarse en la ecuación 2.10, entonces:

$$\frac{1}{2H}\left(\frac{\kappa_0}{3}\dot{\rho} + \frac{2\dot{a}}{a^3}\right) + H^2 = -\frac{\kappa_0}{6}(\rho + 3p).$$

Nuevamente, el término H^2 puede sustituirse utilizando la ecuación 2.9, luego se multiplican ambos lados de la ecuación por $2H$. Desarrollando los términos que queden, se llega fácilmente a la expresión:

$$\dot{\rho} + 3H\left(\rho + p\right) = 0. \tag{2.14}$$

Este resultado se conoce como la *Ecuación de continuidad* o *Ecuación de fluido*, que describe el comportamiento y la evolución del contenido del Universo en función de la densidad y la presión. También suele llamarse la ecuación de conservación de la energía, pues se puede deducir con base en el cambio de energía interna del universo a través de la primera ley de la termodinámica. La ecuación de continuidad es crucial si se quiere inferir cómo se comportará el cosmos en el futuro lejano, o por el contrario, cómo se ha ido transformando desde que ocurrió el Big-Bang. Dicho análisis se puede hacer para cada uno de los componentes ρ_i.

Cuando un sistema es objeto de estudio, resulta de gran utilidad contar con la descripción matemática de sus variables termodinámicas (presión, densidad, volumen, etc). A este tipo de relación se le conoce como *ecuación de estado*. En cosmología se asocian ecuaciones de estado al suponer el contenido del cosmos como un fluido. Esta ecuación ofrece una relación entre la densidad y la presión. La forma más sencilla de obtenerla, es asumir que los componentes se comportan como fluidos perfectos y se describen mediante la siguiente relación barotrópica:

$$p_i = (\gamma_i - 1)\rho_i, \tag{2.15}$$

donde γ_i es característica de cada uno de los componentes. En conjunción con la ecuación de fluido 2.14 y su respectiva ecuación de estado, es posible encontrar una expresión que indique cómo cambia la densidad de cada componente del universo en función del tiempo y el parámetro de expansión adimensional.

Materia

A la materia, que tal y como se mencionó incluye a la materia bariónica y oscura, se le asigna el valor de $\gamma_m = 1$. Esto deja una relación para la presión de $p = 0$. Entonces al sustituir estos valores en la ecuación de fluido 2.14, se obtiene la ecuación diferencial:

$$\dot{\rho}_m + 3H\rho_m = 0.$$

Recordando que $H = \dot{a}/a$ y agrupando las derivadas se tiene:

$$\frac{1}{a^3}\frac{d}{dt}(\rho_m a^3) = 0,$$

$$\rho_m = \frac{c_0}{a^3}, \qquad c_0 = \text{cte}.$$

Al evaluar la densidad para la época actual se obtiene el valor de la constante c_0, entonces la solución queda de la forma:

$$\rho_m = \frac{\rho_{m,0}}{a^3}. \tag{2.16}$$

Radiación

Para las partículas relativistas, se tiene $\gamma_r = 4/3$. De manera análoga al de la materia, la ecuación de fluido entrega la ecuación diferencial:

$$\dot{\rho}_r = 4\frac{a}{\dot{a}}\rho_r = 0.$$

Entonces, resolviendo y evaluando en el tiempo actual para determinar la constante de integración se llega a la ecuación:

$$\rho_r = \frac{\rho_{r,0}}{a^4}. \tag{2.17}$$

Energía oscura

Por otro lado, para la constante cosmológica se toma $\gamma_\Lambda = 0$. De donde se obtiene la ecuación de estado:

$$p_\Lambda = -\rho_\Lambda.$$

Dado, que la densidad es positiva, la relación anterior definiría una presión negativa para la energía oscura. No sólo eso, pues al determinar su ecuación diferencial con ayuda de la ecuación de fluido se tiene:

$$\dot{\rho}_\Lambda = 0.$$

Lo que indica que la densidad correspondiente a la energía oscura no cambia con respecto del tiempo. Evaluando la solución de esta ecuación en los valores hoy en día:

$$\rho_\Lambda = \rho_{\Lambda,0}. \tag{2.18}$$

Así, se ha encontrado una expresión funcional para la evolución del contenido del universo a través del tiempo y en función del parámetro de expansión. Una consecuencia importante de la expansión del universo es que su contenido se ha visto diluido, como se puede apreciar de las expresiones 2.16 y 2.17. No así para el caso de la energía oscura, pues como se puede apreciar de la ecuación 2.18, la densidad de esta es una constante y su presión asociada es por definición negativa.

2.2 Ecuación paramétrica de Friedmann

A partir de la definición de los parámetros adimensionales de densidad (Ecuación 2.12) se puede escribir la ecuación de Friedman 2.9 como:

$$H^2 = \Omega\, H^2 - \frac{k}{a^2}. \tag{2.19}$$

Lo que permite expresar a k en términos de los parámetros de densidad y de H, al evaluarlos en la actualidad se obtiene:

$$k = H_0^2 \left(\Omega_0 - 1\right) a_0^2. \tag{2.20}$$

Así que, con ayuda de la expresión 2.20 se puede escribir la ecuación 2.7 sin depender del parámetro de curvatura de forma explícita:

$$H^2 = \frac{\kappa_0}{3}\rho - \frac{a_0^2}{a^2} H_0^2 \left(\Omega - 1\right), \tag{2.21}$$

recordando que el parámetro de expansión adimensional a vale 1 para la época actual y haciendo algunos despejes se obtiene:

$$\frac{H^2}{H_0^2} = \frac{\rho}{\rho_{c,0}} + \frac{1 - \Omega_0}{a^2}, \tag{2.22}$$

por otro lado, hay que recordar que ρ representa la suma de todas las componentes del universo ρ_i y utilizando las expresiones 2.16, 2.17 y 2.18, encontradas en la sección anterior, se obtiene:

$$\frac{H^2}{H_0^2} = \frac{\Omega_{r,0}}{a^4} + \frac{\Omega_{m,0}}{a^3} + \Omega_{\Lambda,0} + \frac{1 - \Omega_0}{a^2}. \tag{2.23}$$

La ecuación 2.23 se conoce como la *Ecuación paramétrica de Friedmann* y su importancia radica en la posibilidad de relacionar el parámetro de Hubble con el contenido del cosmos representado por los parámetros de densidad normalizados y el parámetro adimensional relacionado con la expansión del universo.

En términos de las observaciones, se mencionó que la expansión del universo afecta en mayor grado a la densidad de radiación comparada con la de materia. La expansión además influye sobre la radiación por medio del corrimiento al rojo (redshift) cosmológico, un fenómeno que hace que la luz tienda hacia longitudes de onda más grandes para la luz observada entre dos puntos en función de la distancia entre ellos y el tiempo en que fue emitida y en el que se observa. Este efecto se puede describir

con la relación:

$$\frac{\lambda_0}{\lambda} = \frac{a_0}{a},$$

con λ_0 la longitud apreciada y λ la que fue emitida originalmente. Además, el corrimiento al rojo z se define como:

$$z = \frac{\lambda_0 - \lambda}{\lambda}.$$

Juntando ambas expresiones, se obtiene que el corrimiento al rojo en términos del parámetro de expansión es:

$$1 + z = \frac{a_0}{a}. \tag{2.24}$$

Usando la expresión para a de la ecuación 2.24, la ecuación paramétrica de Friedmann en función del corrimiento al rojo se escribe como:

$$\frac{H^2}{H_0^2} = \Omega_{r,0}(1+z)^4 + \Omega_{m,0}(1+z)^3 + (1-\Omega_0)(1+z)^2 + \Omega_{\Lambda,0}. \tag{2.25}$$

2.3 Evolución para varias componentes

El análisis en la sección 2.1 fue hecho para componentes individuales en el universo. ¿Qué sucede cuando se consideran todos los componentes interactuando a la vez? En ese caso se debe tomar las respectivas ecuaciones de estado para cada componente, sustituyendo el valor de p_i en la ecuación (2.14). Para cada componente se obtiene un sistema de cuatro ecuaciones diferenciales; tres para los componentes del Universo, además de una correspondiente a $H(t)$, pues también es función del tiempo:

$$\dot{\rho}_i + 3\gamma_i H \rho_i = 0. \tag{2.26}$$

En términos de los parámetros de densidad adimensionales, el sistema (2.26) se puede escribir como:

$$\Omega_i' = 3(\Pi - \gamma_i)\Omega_i, \tag{2.27}$$

con $\Pi = \sum_i \gamma_i \Omega_i$. También se ha realizado el cambio de variable tal que la notación prima significa derivada con respecto al parámetro $N = \ln(a)$. Este sistema se puede resolver tradicionalmente, estableciendo unas condiciones iniciales para los parámetros de densidad de radiación, materia, energía oscura y para la constante de Hubble: $\Omega_{r,0}(t)$, $\Omega_{m,0}(t)$, $\Omega_{\Lambda,0}(t)$, H_0. Así, el sistema cuenta con el conjunto de soluciones:

$$\Omega_r(a) = \frac{\Omega_{r,0}\, a^{-4}}{\Omega_{\Lambda,0} + (1 - \Omega_{\Lambda,0} - \Omega_{m,0} - \Omega_{r,0})\, a^{-2} + \Omega_{m,0}\, a^{-3} + \Omega_{r,0}\, a^{-4}}, \tag{2.28}$$

$$\Omega_m(a) = \frac{\Omega_{m,0}\, a^{-3}}{\Omega_{\Lambda,0} + (1 - \Omega_{\Lambda,0} - \Omega_{m,0} - \Omega_{r,0})\, a^{-2} + \Omega_{m,0}\, a^{-3} + \Omega_{r,0}\, a^{-4}}, \tag{2.29}$$

$$\Omega_\Lambda(a) = \frac{\Omega_{\Lambda,0}}{\Omega_{\Lambda,0} + (1 - \Omega_{\Lambda,0} - \Omega_{m,0} - \Omega_{r,0})\, a^{-2} + \Omega_{m,0}\, a^{-3} + \Omega_{r,0}\, a^{-4}}. \tag{2.30}$$

En adición con la expresión para H dada por la ecuación 2.23. En un universo con las características matemáticas que aquí se asumieron desde el inicio del capítulo, se cuenta con un conjunto de soluciones analíticas para los parámetros de densidad. En general, esto no será posible y el sistema de ecuaciones 2.26 debe tratarse de manera numérica. Debido a eso, en este trabajo se trabajará con dicho sistema a través de las redes neuronales (ver Sección 6.2), aunque antes serán presentados los fundamentos teóricos y matemáticos del aprendizaje profundo.

3 | Fundamentos del aprendizaje profundo

3.1 Breve historia de las redes neuronales artificiales

Desde sus versiones más incipientes, las computadoras han ayudado a llevar a cabo tareas que para un ser humano resultarían muy complicadas o incluso inalcanzables. Pueden realizar cálculos matemáticos inconmensurables o múltiples operaciones simultáneas en cuestión de segundos. Sin embargo, muchos de los comportamientos inteligentes de los seres vivos son difíciles de emular, tales como: el lenguaje, la visión o la capacidad de inferir conclusiones a partir de cierta información. Es aquí donde un conjunto de técnicas y tecnologías encaminadas a reproducir cada vez mejor esos comportamientos entra a escena, la *Inteligencia Artificial*.

Una de las cuestiones más importantes dentro de la inteligencia artificial radica en la forma que el cerebro procesa la información. Para comprender esto fue necesario pensar a nivel celular, proponiendo modelos para las neuronas individuales que luego evolucionaron a sistemas más complejos de redes, redes neuronales. Así las Redes Neuronales Artificiales comenzaron en 1943 cuando se propuso el primer modelo lógico computacional que emulaba el funcionamiento de una neurona [26]. Este modelo sólo era capaz de resolver unas pocas puertas lógicas linealmente separables, y además era necesario proporcionar una determinada cantidad de información sobre el problema. Unos años más tarde, en 1957 Frank Rosenblatt [27] propuso el precursor más cercano

a las redes neuronales actuales, el modelo computacional conocido como *perceptron*.

Ese nuevo modelo proponía cambios en el proceso de aprendizaje, a diferencia de su predecesor, la neurona era capaz de aprender por sí misma a partir de las señales de entrada. Sin embargo, en 1969 Marvin Minsky y Seymour Papert, famosos investigadores de la Inteligencia Artificial, pusieron de manifiesto la incapacidad del perceptrón para resolver algunos problemas no lineales, como el conocido problema de clasificación denominado *Organizador exclusivo* (XOR) [28]. Debido a esto, la comunidad científica y tecnológica perdió el interés por las redes neuronales y tuvieron que pasar muchos años para que esta situación cambiara. Esta etapa de estancamiento se conoce como el *invierno de la inteligencia artificial*. La curiosidad por las redes neuronales se recuperó en la década de 1980, impulsada por Geoffrey Hinton y sus colegas, que propusieron el algoritmo de *retropropagación* con el que resolvieron algunas limitaciones computacionales de las redes neuronales [29]. Sin embargo, fue hasta el siglo XXI, debido a los avances informáticos, que las redes neuronales recuperaron gran relevancia, incluso nació una nueva disciplina científica: el *Aprendizaje Profundo*, centrada exclusivamente en el estudio de las Redes Neuronales Artificiales.

El aprendizaje automático (Machine Learning) es una rama de la Inteligencia Artificial dedicada al modelado matemático de los datos, algunas de sus principales aplicaciones se encuentran en tareas de clasificación, regresión y la agrupación. Existen tres categorías principales en las que se divide el aprendizaje automático: aprendizaje supervisado, aprendizaje no supervisado y aprendizaje por refuerzo. En el aprendizaje supervisado, el conjunto de datos de entrenamiento contiene los valores de las entradas y de los valores que se desea predecir. Si en el conjunto de datos de entrenamiento no se encuentran los valores que se desea modelar, se utilizan técnicas de aprendizaje no supervisado. En el aprendizaje por refuerzo, el sistema debe aprender por sí mismo la mejor estrategia para obtener la mayor recompensa con base en comportamientos deseados o no deseados en determinado entorno.

Así pues, el aprendizaje profundo es considerado un subcampo del aprendizaje automático que puede realizar un gran abanico de tareas, tales como: regresión, clasificación, procesamiento de imágenes y vídeos, procesos generativos, series temporales,

entre otros.

En las últimas décadas se ha popularizado la implementación de modelos basados en aprendizaje profundo en una gran cantidad de áreas distintas, principalmente por la democratización del acceso a los sistemas de cómputo: como los ordenadores de mesa, cada vez más baratos y eficientes o la creciente oferta de las conocidas GPU (graphics processing unit). Finalmente, las aplicaciones del aprendizaje profundo han repercutido en gran número de áreas distintas, como: las finanzas [30], las redes sociales [31, 32], la medicina [33] e incluso los videojuegos [34]. Así se acumulan esfuerzos que ayudan a cerrar un poco la gran brecha entre las computadoras y los seres vivos en cuestión de inteligencia.

3.2 El perceptrón

El Perceptron es la unidad fundamental de las redes neuronales, se trata del modelo matemático de una neurona biológica. Las neuronas del cerebro reciben señales provenientes de sus vecinas a través de la sinapsis, esta información viaja en forma de impulsos eléctricos y estimula a la célula haciendo que esta a su vez mande señales a otras neuronas, siempre y cuando el estímulo que recibe logra sobrepasar un determinado valor umbral [35]. De manera análoga, el perceptrón recibe la señal de n entradas numéricas, $x_1, x_2, \cdots, x_n$, cada una tiene asociado un coeficiente w_j llamado *peso* (Figura 3.1). El perceptrón realiza una transformación lineal a las entradas mediante los pesos asociados y una constante b, denominada *sesgo*. Esta transformación entre pesos y entradas se conoce como *suma ponderada* z:

$$z = \sum_{j=1}^{n} (x_j w_j) + b. \tag{3.1}$$

El sesgo indica el valor que debe tomar la salida cuando todas las x_i son nulas y los pesos determinan la influencia que su entrada correspondiente tendrá sobre la salida del perceptrón. Al igual que las neuronas biológicas, es necesario hacer una modulación sobre la señal entrante z, en el aprendizaje profundo esa tarea se realiza

a través de las *funciones de activación*, cuyas características serán expuestas en la Sección 4.1. Sin embargo, se puede decir que la salida del perceptrón viene dada por la función de activación (denotada por σ) aplicada a la suma ponderada:

$$a = \sigma(z). \tag{3.2}$$

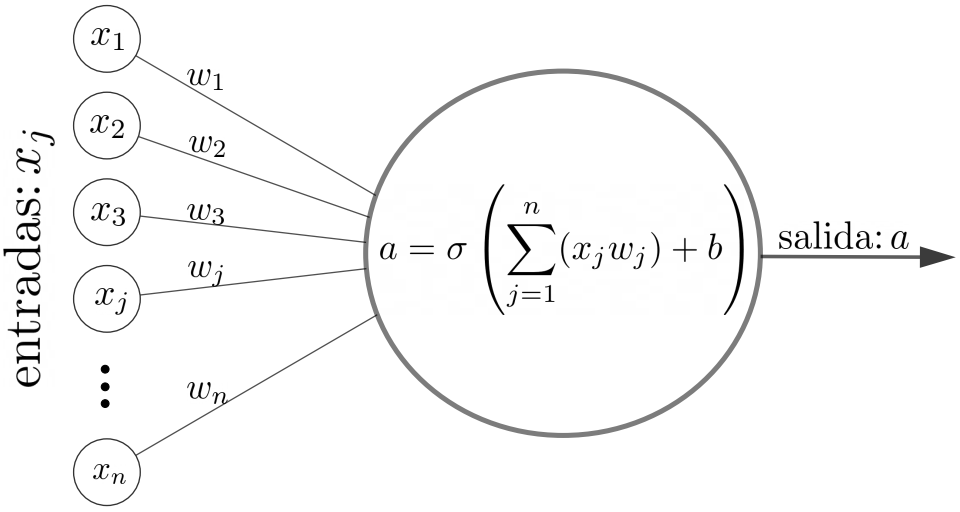


Figura 3.1: Esquema de un perceptrón, el modelo en aprendizaje profundo de una neurona biológica.

El perceptrón procesa las entradas junto con sus pesos asociados y devuelve una salida modulada por la función de activación aplicada a z. Esta estructura computacional también es comúnmente llamada *nodo* o *neurona*, y servirá como ladrillo para construir redes neuronales profundas (múltiples arreglos de nodos). Este modelo puede realizar algunas tareas lineales simples, como procesos de clasificación binaria o regresiones lineales. Para mostrar cómo se lleva a cabo el proceso de aprendizaje del perceptrón, que después podrá generalizarse a las redes profundas, a continuación se muestra un ejemplo de regresión lineal.

3.3 Regresión lineal

Dentro de los tipos de regresión estándar, la lineal es posiblemente las más usual y la más sencilla de todas. Fue estudiada por Gauss y Legendre en el siglo XIX y es ampliamente utilizada en física y matemáticas. Suponiendo que se tiene una variable y como función de n variables x_i independientes entre sí, existirá una relación lineal entre las variables si y puede escribirse como una suma ponderada de las x_i.

$$y = x_1 w_1 + x_2 w_2 + \cdots + x_n w_n + b. \tag{3.3}$$

Esto para algunos coeficientes constantes w_i y b. Por lo tanto, para determinar el valor de y en función del conjunto $\{x_i\}$ es imperativo determinar los valores de $\{w_i\}$ y b, y de eso se encarga el proceso de regresión lineal. Si se cuenta con un conjunto de datos de m elementos con n variables independientes x, estos se pueden organizar como un sistema de m ecuaciones con n variables:

$$\begin{aligned}
y_1 &= x_{11} w_1 + x_{12} w_2 + \cdots + x_{1n} w_n + b, \\
y_2 &= x_{21} w_1 + x_{22} w_2 + \cdots + x_{2n} w_n + b, \\
&\vdots \\
y_m &= x_{m1} w_1 + x_{m2} w_2 + \cdots + x_{mn} w_n + b.
\end{aligned}$$

Para simplificar, esto se puede escribir en notación matricial:

$$Y = XW + b. \tag{3.4}$$

En aprendizaje automático, el conjunto de variables independientes X suele llamarse *características*, *atributos* o *muestras*; W se denomina *matriz ponderada* o *matriz de pesos*; y Y, *etiquetas* o *clases*. Ahora, si se define la función de activación como la identidad $\sigma(x) = Id(x)$, es posible reproducir la ecuación (3.4) con el perceptrón. Más aún, este podrá realizar una regresión lineal para determinar el valor de W y b

que mejor representen a los conjuntos X, Y.

Para comprender qué tan bien el modelo dado por el perceptrón describe los datos reales, es necesario medir la diferencia entre los valores predichos y los esperados. Para esto se suele recurrir a una función que sirva como métrica entre el modelo y los datos, la función utilizada para cuantificar esta diferencia se conoce como *Función de pérdida* o *Función de coste*. Esta función dependerá de los parámetros w_j y b, y debe minimizarse tanto como sea posible para obtener una predicción aceptable. Existen muchas funciones de pérdida que pueden llevar a cabo esta tarea y elegir cuál usar depende del contexto del problema. Por ejemplo, el error cuadrático medio (MSE) es una de las funciones más socorridas en los problemas de regresión. Se define como:

$$C = \frac{1}{2}(\underbrace{y_j}_{real} - \underbrace{(x_{j1}w_1 + x_{j2}w_2 + \cdots + x_{jn}w_n + b)}_{modelo})^2.$$

Dado que es necesario considerar el error aportado por el conjunto de datos en su totalidad, se debe sumar sobre todos los elementos j. En notación matricial se tiene:

$$C(W, b) = \frac{1}{2m}\|Y - (XW + b)\|^2. \tag{3.5}$$

Una vez que se han encontrado $\{w_i\}$ y b que minimizan la ecuación (3.5), existe entonces un modelo lineal que se ajusta a los datos X, Y de buena manera. En otras palabras, la función de coste $C(W, b)$ se aplica a la salida del perceptrón para encontrar la matriz de pesos W y el sesgo b que optimizan el modelo.

3.4 Algoritmo Descenso del Gradiente

Existen muchos métodos utilizados en la optimización de funciones; sin embargo, el más popular en el aprendizaje profundo es el conocido como *Descenso del Gradiente*. Este algoritmo es versátil y puede generalizarse fácilmente a funciones multivariables. Dada una función escalar diferenciable $f : \mathbb{R}^n \longrightarrow \mathbb{R}$, el vector cuyos componentes son las derivadas parciales de f, se llama **Gradiente**.

$$\nabla f(x) = \left(\frac{\partial f}{\partial x_1}(x), \frac{\partial f}{\partial x_2}(x), \cdots, \frac{\partial f}{\partial x_n}(x) \right).$$

El vector gradiente tiene algunas propiedades geométricas interesantes:

- El vector gradiente $\nabla f(x_0)$ es ortogonal a la curva de nivel $f(x) = k$ en el punto x_0.
- $\nabla f(x)$ apunta en la dirección de mayor crecimiento para f.
- Por el contrario, $-\nabla f(x)$ apunta en la dirección de mayor decrecimiento.

El algoritmo de descenso del gradiente se basa en la tercera propiedad. Entonces, dada una función escalar f la cual tiene un mínimo en x_0 y un punto x se encuentra en una determinada región del dominio: dar un paso en la dirección $-\nabla f(x)$ es estar un paso más cerca del punto crítico x_0. Con un número suficiente de iteraciones, se puede encontrar el valor para el cual se minimiza f. La siguiente ecuación sintetiza el mecanismo de descenso de gradiente:

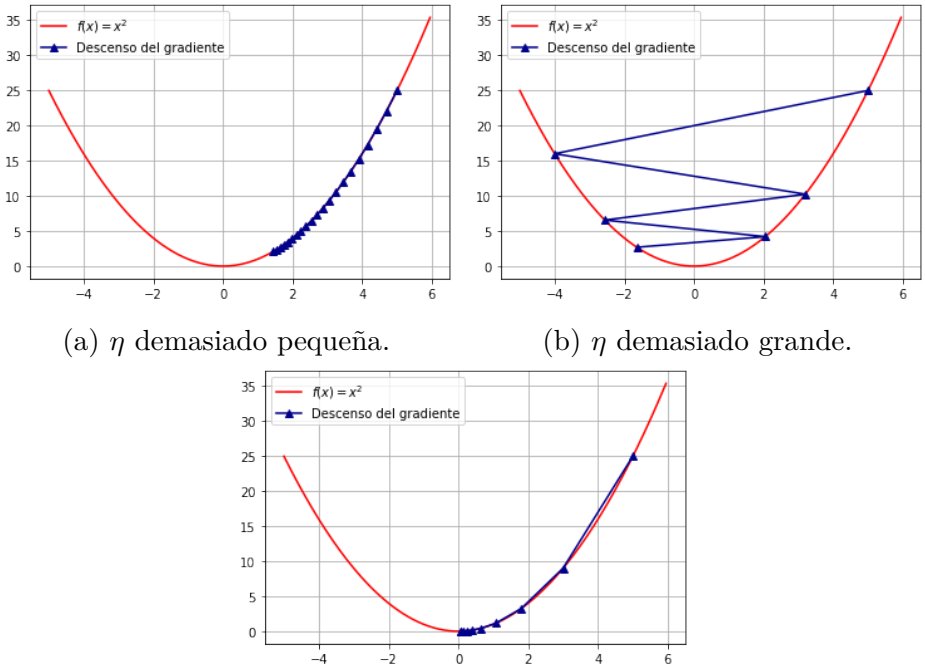
$$x' \longrightarrow x - \nabla f(x), \tag{3.6}$$

donde x' es el nuevo punto que tomará el algoritmo y x el punto actual.

Un parámetro importante utilizado en este algoritmo es la *tasa de aprendizaje*. Este parámetro determina el tamaño del paso que se toma cada vez que realiza una iteración, por lo que este valor es muy importante, ya que influye en si el algoritmo es capaz de converger o no al valor objetivo de manera adecuada. La tasa de aprendizaje, comúnmente denotada por η, suele ser un número real constante y positivo (comúnmente $\eta \in [0, 1]$) por el cual se multiplica el gradiente al aplicar la ecuación 3.6, por lo que las iteraciones se harán siguiendo la ecuación:

$$x' \longrightarrow x - \eta \nabla f(x). \tag{3.7}$$

¿Qué valor debería tomar η? Dependerá de la función a minimizar, para diferentes tipos de problemas se suelen usar diferentes valores. También existen casos en los que



(a) η demasiado pequeña. (b) η demasiado grande.

(c) η óptima.

Figura 3.2: Tres selecciones diferentes de la tasa de aprendizaje (líneas azules) y su impacto en el descenso del gradiente cuando se aplica para minimizar una función (línea roja sólida).

se puede utilizar una tasa de aprendizaje dinámica, cuyo tamaño cambia y se encoge a medida que se acerca al mínimo de la función. Entonces, el tamaño del paso en cada iteración depende de dos factores: la norma del vector de gradiente y el valor η. Cuanto más cerca del punto crítico donde se minimiza f, el tamaño del paso tiende a acortarse; aunque η sea constante, ya que en el punto crítico la norma del gradiente tiende a cero. Hay tres casos a considerar de la magnitud de la tasa de aprendizaje:

- η **demasiado pequeña**: Si η es muy pequeña, el tamaño del paso será demasiado corto, como consecuencia podría aumentar el tiempo total de cálculo en gran medida o incluso podría fallar al converger al punto deseado (ver Figura 3.2a).

- η **demasiado grande**: Una tasa de aprendizaje muy grande puede causar que el gradiente explote y diverja, o el algoritmo puede pasar por alto el mínimo y sobrepasarlo (ver Figura 3.2b).

- η **óptima**: una tasa de aprendizaje adecuada permite que el algoritmo pueda converger al mínimo de la función en un número razonable de iteraciones, lo que también reduce el tiempo de cálculo del proceso (Figura 3.2c).

La esencia de este algoritmo reside entonces en encontrar el vector gradiente de la función, y en primera instancia, las derivadas parciales de esta. Desde el punto de vista computacional existen diversas maneras de obtener dichas derivadas: pueden ser provistas al algoritmo si se conocen de manera analítica, calculadas con librerías de matemática simbólica, utilizando métodos numéricos, o el famoso método de diferenciación automática.

Ejemplo:

Para dejar más claro el método del descenso del gradiente, se aplicará a una función escalar $f(x_1, x_2) = x_1^2 + x_2^2$ utilizando el método numérico de diferencias finitas para computar las derivadas parciales. El pseudo código para implementar el descenso del gradiente se puede sintetizar en tres pasos:

1. Derivadas parciales: Calcular las derivadas parciales de cada variable x_k y evaluarlas en un punto de partida aleatorio x_0 en el dominio de la función, $\frac{\partial f}{\partial x_k}(x_0)$.

2. El gradiente: Se debe construir el gradiente de la función. Es recomendable definir un valor máximo para la norma del gradiente, pero conservando su dirección, para evitar divergencia al hacer las iteraciones.

3. Descenso de gradiente: cuando la ecuación 3.7 se ha aplicado al punto inicial x_0, el proceso se repite para el nuevo punto hasta que se esté lo suficientemente cerca del punto donde f alcanza su mínimo.

Aplicando el descenso del gradiente a la función para obtener su mínimo, con una tasa de aprendizaje $\eta = 0.1$ y un punto inicial $x_0 = (x_{1,0}, x_{2,0}) = (-10, 2.8)$. En la Figura 3.3 se puede apreciar la ruta que realiza el algoritmo durante el proceso, a través de los pasos sobre las curvas de nivel de f hasta llegar al punto donde se minimiza, en este caso el óptimo está en $(x_1, x_2) = (0, 0)$.

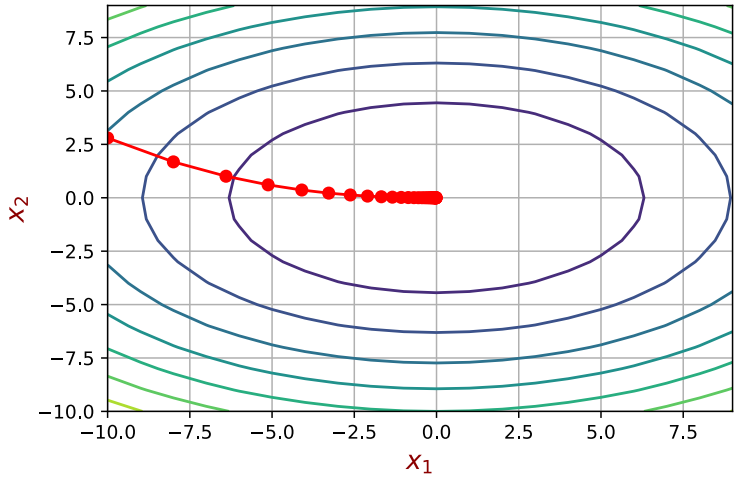


Figura 3.3: La línea roja muestra el camino recorrido por el descenso de gradiente en el dominio de f para encontrar su mínimo. Las elipses corresponden a las curvas de nivel de f.

Tanto en este ejemplo como en la figura 3.2 se puede observar que el descenso del gradiente se aplicó a un par de funciones y en ambos casos se encontraron sus mínimos, aunque esto no siempre será posible para cualquier función. En general, habrá dificultades con funciones que tienen mínimos locales o que son no convexas. Se puede garantizar la convergencia del algoritmo si f cumple un par de condiciones [36]:

1. Ser convexa.

2. Que su vector gradiente sea Lipschitz continuo, esto es:
   $\forall(x, y) \in \text{Dom}_f$, $\exists L \in \mathbb{R}^+$ t.q $\|\nabla f(x) - \nabla f(y)\| < L\|x - y\|$.

El descenso de gradiente se puede aplicar a muchas funciones de costo que satisfagan estas condiciones, lo que asegura obtener el punto crítico buscado. Un ejemplo de una función que satisface las características anteriores es el error cuadrático medio (MSE). Por lo tanto, podemos aplicar la ecuación (3.5) al perceptrón y luego minimizarla con la ayuda del descenso del gradiente.

3.5 Ejemplo: modelado de la ley de Hubble con el perceptron

En este ejemplo se presenta una regresión lineal utilizando datos cosmológicos, los cuales permiten ajustar un valor observacional del parámetro de Hubble. En 1929, Edwin Hubble descubrió que las galaxias se alejan unas de otras con una velocidad proporcional a la distancia que las separa [37], la constante de proporcionalidad de esta relación se conoce como el parámetro de Hubble H_0, que cumple con la siguiente relación:

$$v = H_0 r, \tag{3.8}$$

esta ecuación se conoce como *ley de Hubble* [38]. Cabe mencionar que esta relación es válida sólo para objetos con bajo corrimiento al rojo, pues como se mencionó en el capítulo 2, el parámetro de Hubble no es constante y depende del corrimiento al rojo y de los parámetros de densidad asociados al contenido del universo. También es natural deducir que, según la ecuación 3.8, dada una distancia r lo suficiente grande es posible que la velocidad de recesión alcance o incluso supere el valor de la velocidad de la luz c. Esto podría causar conflicto con el bien conocido hecho de que ningún objeto con masa puede viajar con una rapidez igual o superior a la de la luz, pero hay que entender que en la expansión del universo no hay cambio de posiciones entre las galaxias. Ningún objeto puede moverse en el espacio con velocidad mayor a la de la luz, pero en este caso es el propio espacio el que se está expandiendo entre ambos objetos, por lo que la relatividad especial no ha sido violada.

Dicho lo anterior, existen diferentes formas de determinar la distancia entre objetos cosmológicos, una de ellas es la detección de supernovas de tipo Ia y su brillo máximo. Aquí se toman los datos de 36 de estas supernovas (Figura 3.4) y se relacionan su distancia y velocidad de recesión al igual que Hubble en su momento. Los datos correspondientes se pueden consultar con más detalle en [39].

El parámetro H_0 puede interpretarse como la rapidez a la que se expande el universo y es posible determinarlo al aplicarse una regresión lineal a los datos obser-

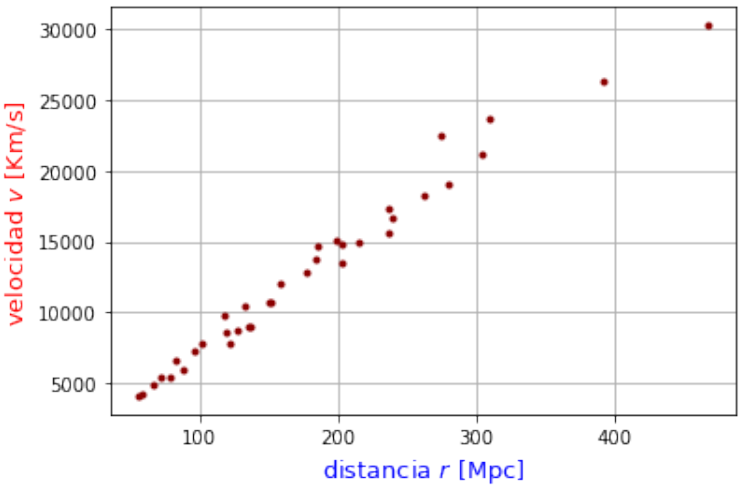


Figura 3.4: Datos observacionales correspondientes a 36 supernovas tipo Ia, que cumplen con la ley de Hubble.

vacionales. Para poder ajustarlos con el perceptrón, se asumirá que la relación entre las variables está dada por la ecuación:

$$v = H_0 r + b. \tag{3.9}$$

Los datos de r serán procesados por el perceptrón, generando una salida que luego será comparada con los valores esperados para v con ayuda de la función de coste MSE (ecuación 3.5), de tal suerte que $C = C(H_0, b)$. Posteriormente, se aplicará el descenso del gradiente para minimizar la función de coste, obteniendo así los parámetros para los que el perceptrón se ajusta de mejor manera a los datos observacionales.

Aquí, H_0 y b son el peso y sesgo del perceptrón respectivamente. Y al proceso donde el descenso del gradiente actúa sobre la función de coste aplicada al perceptrón, modificando así sus parámetros en cada iteración, se le conoce como *proceso de aprendizaje*. Cada una de esas iteraciones es denominada *época*.

En la Figura 3.5, se pueden ver algunos pasos del proceso de aprendizaje realizado por el perceptrón, en los que se tomaron diferentes valores de H_0, de manera que la diferencia entre los datos y el modelo fue cada vez menor. Finalmente, los parámetros que optimizan la regresión lineal de los datos de Hubble, dados por el perceptrón son: $H_0 = 68.00\,\mathrm{km\ s^{-1}Mpc^{-1}}$, $b = 567.00\,\mathrm{km\ s^{-1}}$, con $1\mathrm{pc} = 3.08 \times 10^{18}\,\mathrm{m}$. Este resultado

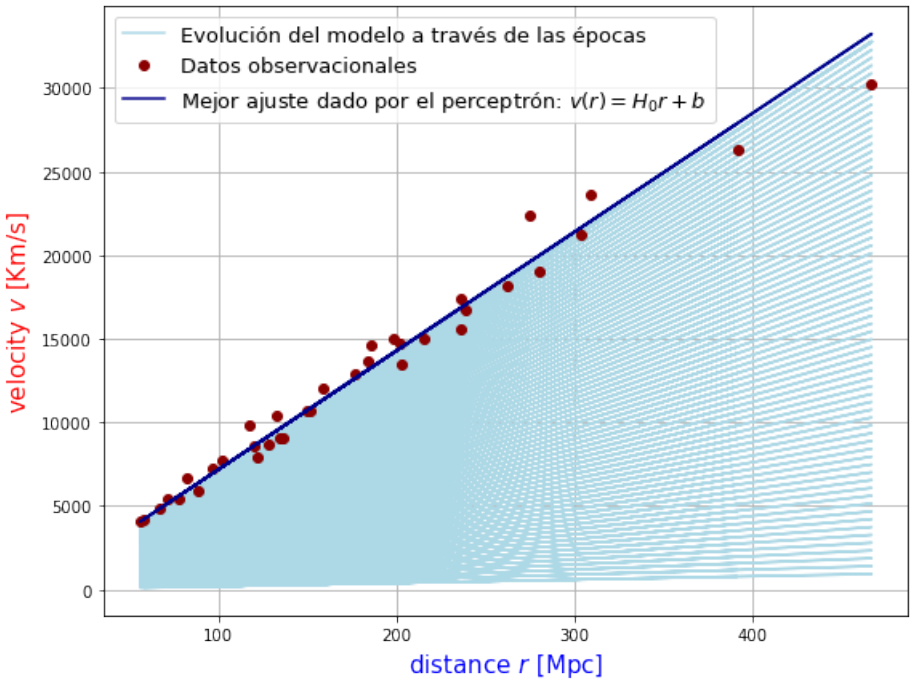


Figura 3.5: Ajuste de datos por el perceptrón. Cada línea azul clara es generada en una época del entrenamiento, la línea azul oscura corresponde al valor mínimo para el MSE y, por tanto, a la configuración del perceptrón más aceptable.

se puede comparar con el obtenido por [39] de $H_0 = 71 \pm 2_r \pm 6_s \,\mathrm{km\,s^{-1}Mpc^{-1}}$, donde los subíndices r y s representan el error aleatorio y sistemático respectivamente. Las diferencias en el resultado se deben principalmente a los ajustes realizados por los autores tales como correcciones de metalicidad: en este caso sólo se realizó un ajuste lineal simple. Finalmente, el resumen del proceso de aprendizaje del perceptrón puede esquematizarse en el siguiente algoritmo:

Algoritmo 1: Aprendizaje de una neurona

Datos: $X = \{x_1, ..., x_n\}$, $Y = \{y_1, ..., y_n\}$.
Generar valores aleatorios para w, b.
para *i in range*(épocas) **hacer**
  para $x_i \in X$ y $y_i \in Y$ **hacer**
    - Generar una salida a_i.
    - Actualizar los parámetros:
      $w_i \longleftarrow w_i - \Delta w_i$,
      $b_i \longleftarrow b_i - \Delta b_i$.
      con $\Delta w_i = \eta(a_i - y_i)\sigma'(z)x_i$,
      $\Delta b_i = \eta(a_i - y_i)\sigma'(z)$,
      - η: tasa de aprendizaje.
      - y_i: valor real de $f(x_i)$.
      - a_i: salida o predicción del perceptrón.
      - σ': derivada de la función de activación.
  fin
fin

4 | Redes neuronales profundas

Una vez entendido el funcionamiento del perceptrón, es viable introducir algunos conceptos de redes neuronales profundas. Los mecanismos de funcionamiento y aprendizaje de una sola neurona se pueden generalizar a muchas de ellas, las redes neuronales profundas. La organización más sencilla de este tipo de redes se conocen como *Perceptrones multicapa* (MLP), que constan de varios perceptrones o nodos interconectados entre sí a través de diferentes arreglos, llamados capas. Cada MLP consta de una capa de entrada; al menos una capa intermedia, a menudo llamadas *capas ocultas*; y una capa de salida (Figura 4.1). La capa de entrada de una RNA corresponde a los datos que se procesarán a través de capas posteriores y el número de nodos en esta capa debe coincidir con el número de variables independientes (características o atributos) del conjunto de datos. Al igual que el perceptrón, la capa de salida debe compararse con las variables dependientes (o clases) con los que se esté trabajando, que son las que se deben modelar o predecir.

Las conexiones entre un nodo y otro tienen un peso asociado w_{ij}^l, y un sesgo b_j^l, donde l representa el número de capa; el peso conecta el nodo i en la capa $l-1$, con el nodo j en la capa mencionada l. En consecuencia, el nodo j en la capa l tendrá una salida a_j^l. La suma ponderada de cada nodo contiene información de los pesos, sesgos y entradas de cada capa, y se calcula de la siguiente manera:

$$z_j^l = \sum_i a_i^{l-1} w_{ij}^l + b_j^l. \tag{4.1}$$

Entonces, las funciones de activación σ se aplican a la suma ponderada (ecuación.

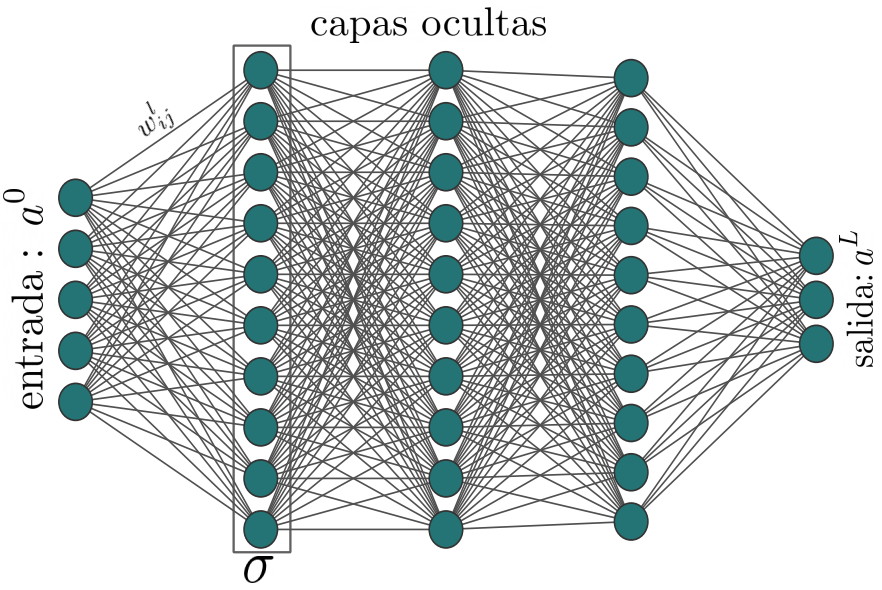


Figura 4.1: Estructura de una red neuronal profunda o perceptrón multicapa. Los componentes de esta arquitectura (círculos verdes) se tratan de perceptrones. La información se transmite desde la capa de entrada a^0, pasando por las capas ocultas (capas entre la entrada y la salida) y concluyen en la capa de salida a^L. La función de activación σ se aplica por cada capa. El número de nodos de entrada y salida se define por el número de variables a tratar en el conjunto de datos.

4.1) y se denotan por:

$$a^l_j = \sigma(z^l_j). \tag{4.2}$$

El uso de la notación matricial permite establecer la estructura del MLP de una forma más sencilla. Sea L la última capa, entonces la salida del MLP es el vector que toma cada nodo a^L_j como entradas:

$$\begin{pmatrix} a^L_1 \\ a^L_2 \\ \vdots \\ a^L_m \end{pmatrix} = a^L. \tag{4.3}$$

Así, es posible denotar a las capas intermedias recursivamente como vectores a^l. Para cada capa, también es posible construir una matriz de pesos W^l y denotar su suma

ponderada z^l como sigue:

$$z^l = a^{l-1}W^l + b^l. \tag{4.4}$$

Obsérvese que las entradas de la matriz de pesos asociada, con número de capa l, se definen como

$$[W^l]_{ij} = w^l_{ij},$$

con b^l el vector que contiene los sesgos de cada nodo de la capa l. A continuación, es necesario aplicar la función de activación σ como se indica en la ecuación 4.2:

$$a^l = \sigma(z^l). \tag{4.5}$$

4.1 Funciones de Activación

Las redes neuronales artificiales son conocidas como aproximadores universales [40]. El teorema de aproximación universal demuestra que, dada una función continua en un conjunto compacto contenido en un espacio $n-$dimensional $f : \mathbb{R}^n \longrightarrow \mathbb{R}^m$, entonces existe una red neuronal con una capa oculta (al menos) cuya función de activación sea no lineal tal que esta aproxima a la función f con cualquier grado de precisión deseado, siempre y cuando se cuente con un número suficiente de nodos en la capa oculta.

El rol de las funciones de activación es fundamental para el aprendizaje profundo: sin ellas sólo se podrían representar transformaciones lineales, sin importar cuántas capas y nodos ocultos se utilicen. La elección de las funciones de activación depende de cada caso, del problema que se aborda y del comportamiento que se requiere de cada capa de la red. Hay varias funciones de activación que cumplen las características de ser no lineales y continuas, pero algunas han sido estudiadas y aplicadas más que otras. A continuación, se muestran algunos ejemplos populares:

- **ReLU**.- *Unidad Lineal Rectificada* (Figura 4.2a) es la función de activación más

popular y sencilla. Dado un número x, entonces

$$\text{ReLU}(x) = \max\{0, x\}. \tag{4.6}$$

Proporcionando una transformación no lineal muy simple sobre $\mathbb{R}$. La función ReLU retiene sólo los elementos positivos y descarta todos los negativos estableciendo $x < 0$ en 0. Aunque su derivada es indefinida cuando $x = 0$ (Figura 4.2b), no es necesario preocuparse por ello porque los valores de entrada difícilmente serán todos nulos al mismo tiempo.

$$\frac{d}{dx}\text{ReLU}(x) = \begin{cases} 0 & \text{si } x < 0, \\ 1 & \text{si } x > 0. \end{cases} \tag{4.7}$$

Las derivadas de las funciones de activación son una parte relevante del proceso de aprendizaje (como se verá en la Sección 4.3). Por lo tanto, es importante considerar sus propiedades.

- **Sigmoide** .- La función sigmoide (Figura 4.2c) mapea la línea real al intervalo $(0, 1)$. El comportamiento de esta función se definió teniendo en cuenta el comportamiento de las neuronas reales, que reciben estímulos y los comunican a través de pulsos. La función sigmoide mapea la recta real de la siguiente manera: los x muy negativos a cero y si x tiende a infinito, su imagen se aplastará a 1; es decir, emula lo mejor posible una función escalonada 0 o 1. Se define como:

$$\text{sigmoide}(x) = \frac{1}{1 + e^{-x}}, \tag{4.8}$$

y su derivada (Figura 4.2d) alcanza su máximo cuando $x = 0$, cuando se aleja de este valor tiende a cero.

$$\frac{d}{dx}\text{sigmoide}(x) = \frac{e^{-x}}{(1 + e^{-x})^2} = \text{sigmoide}(x)(1 - \text{sigmoide}(x)). \tag{4.9}$$

La segunda definición es mejor en términos computacionales.

- **Tangente hiperbólica** .- Esta función (Figura 4.2e) tiene un comportamiento similar a la función sigmoide, de la misma manera, mapea el conjunto de números reales en un intervalo $(-1, 1)$. Se puede ver que, en puntos cercanos a 0, la función tangente hiperbólica ($tanh$) tiene un comportamiento casi lineal y es simétrica con respecto al eje Y.

$$\tanh(x) = \frac{1 - e^{-2x}}{1 + e^{-2x}}. \tag{4.10}$$

  Con respecto a su derivada (Figura 4.2f), el comportamiento es similar al de la derivada de sigmoide, y también se puede expresar en términos de la propia función.

$$\frac{d}{dx}\tanh(x) = 1 - \tanh(x)^2. \tag{4.11}$$

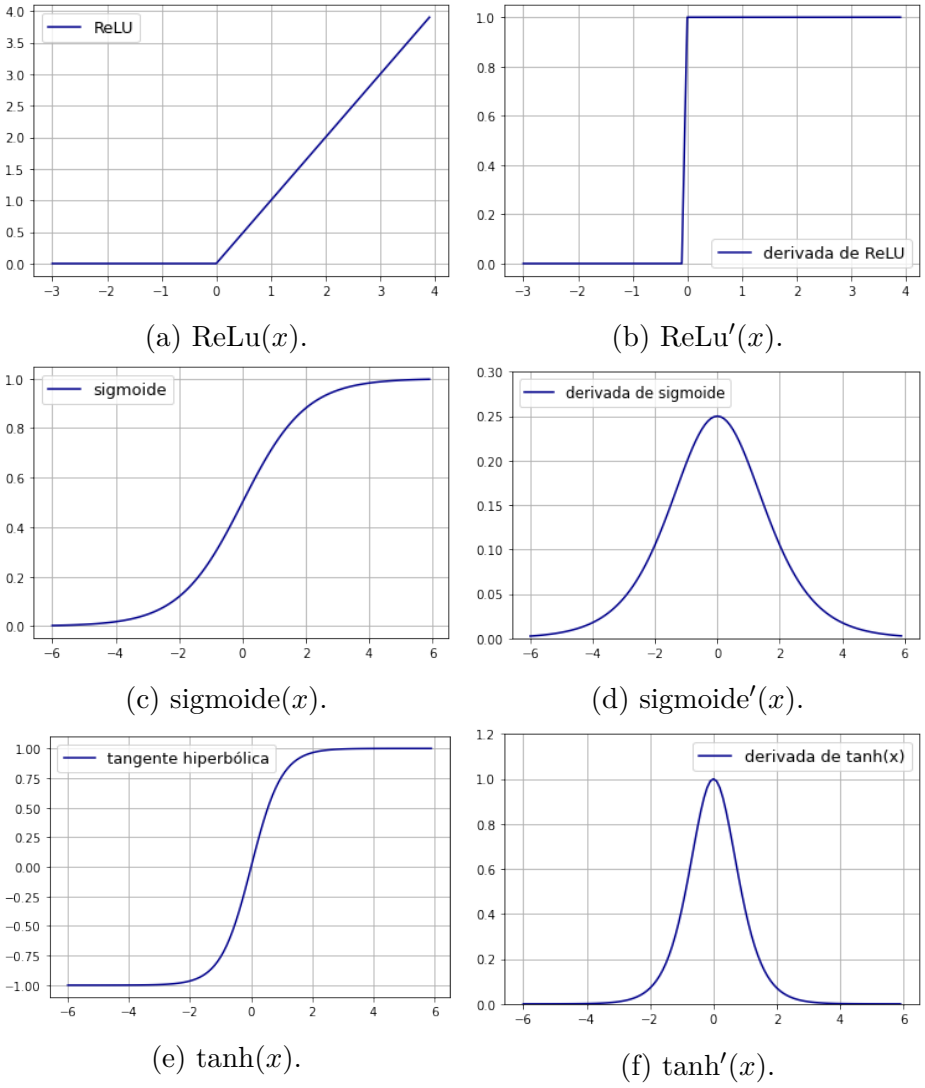
4.2 Propagación hacia delante

El proceso por el cual se transmite la información desde las entradas de la red hasta generar una salida o predicción en la última capa se conoce como *Propagación hacia adelante*. Un perceptrón multicapa se puede considerar como una función $f : a^0 \in \mathbb{R}^n \longrightarrow a^L \in \mathbb{R}^k$. En esta función, las capas ocultas participan de manera implícita en la generación de la salida de la red. La propagación hacia adelante de un MLP se representa en el siguiente diagrama:

$$a^0 \longrightarrow W^1 a^0 + b^1 \xrightarrow{\sigma} a^1 \longrightarrow \cdots \longrightarrow a^{L-1} \longrightarrow W^L a^{L-1} + b^L \xrightarrow{\sigma} a^L. \tag{4.12}$$

Por ejemplo, suponga que la estructura de una red neuronal consta de una sola capa oculta, con un número j de neuronas y una función de activación no lineal σ. Usando las ecuaciones 4.4 y 4.5, podemos calcular la salida de la capa oculta a^1 de la siguiente manera:

$$z^1 = a^0 W^1 + b^1.$$



(a) ReLu(x). (b) ReLu$'(x)$.

(c) sigmoide(x). (d) sigmoide$'(x)$.

(e) tanh(x). (f) tanh$'(x)$.

Figura 4.2: Algunas funciones de activación recurrentes.

Luego, se construye la matriz de pesos W^1 tomando como columna j el vector de pesos que ingresa en el nodo j, así en este caso $W^1 \in \mathrm{M}_{n\times h}$ y $b^1 \in \mathbb{R}^h$.
Ahora, elegimos la función de activación y la aplicamos a cada entrada de z^1:

$$\sigma(z^1) = a^1 \in \mathbb{R}^h.$$

Finalmente aplicamos este mismo proceso para las siguientes capas ocultas, si las

hay. En este caso, para la capa de salida:

$$z^2 = a^1 W^2 + b^2,$$

con $W^2 \in \mathrm{M}_{h\times k}$ y $b^2 \in \mathbb{R}^k$, si tomamos la identidad como activación para esta capa, tendremos que la salida del MLP está dada por:

$$Id(z^2) = a^2 \in \mathbb{R}^k.$$

Este proceso de transmisión de información desde la primera capa a la última es un ejemplo de *propagación hacia adelante*. La salida a^2 es la predicción de este perceptrón multicapa. La propagación hacia adelante es el primer paso del entrenamiento de una red neuronal; sin embargo, son necesarios otros mecanismos para que las predicciones, que serán evaluadas en la función de costo, puedan actualizarse para acercarse cada vez más a los valores objetivo.

4.3 Algoritmo de retropropagación

Al igual que con el perceptrón, en las redes neuronales multicapa se utilizan funciones de costo como métrica entre la salida de la red y los valores esperados. Como ya se mencionó, existen distintas funciones de coste utilizadas en la gran variedad de tareas que una red neuronal profunda puede realizar. En particular, el error cuadrático medio se puede aplicar a los perceptrones multicapa para problemas de regresión, así se puede comparar la salida del MLP a^L con el valor(es) esperado Y:

$$C = \frac{1}{2}\|Y - a^L\|^2 = \frac{1}{2}\sum_j \left(Y_j - a_j^L\right)^2. \tag{4.13}$$

Y como es de esperar, la ecuación 4.13 también debe ser minimizada para obtener un modelo aceptable por parte de la RNA. Para optimizar la ecuación se hará uso del ya conocido descenso del gradiente. En este caso, los parámetros que deben ser actualizados en cada iteración son los pesos w_{ij}^k y sesgos b_j^l, puesto que los cambios

de cada uno de estos parámetros afectan a la salida de la red y por ende a la función de coste. Las actualizaciones se harán utilizando la notación matricial, por lo que los cambios en las iteraciones serán sobre las matrices W^l y b^l.

Un par de ventajas de este algoritmo son que indican cómo se deben actualizar las variables de la función de costo a lo largo del proceso de aprendizaje además de que no incluye derivadas de orden superior, facilitando su implementación. Retomando la esencia del descenso del gradiente, y como en este caso se tienen múltiples variables influyendo sobre la función de costo, la ecuación 3.7 se reescribe como:

$$W^l \longrightarrow W^l - \eta \nabla_{W^l} C = W^l - \eta\, \partial_{W^l} \left(\frac{1}{2} \sum_j \left(Y_j - a_j^L \right)^2 \right), \tag{4.14}$$

$$b^l \longrightarrow b^l - \eta \nabla_{b^l} C = b^l - \eta\, \partial_{b^l} \left(\frac{1}{2} \sum_j \left(Y_j - a_j^L \right)^2 \right). \tag{4.15}$$

Donde η vuelve a ser la tasa de aprendizaje. El problema aquí radica en que la función C no depende explícitamente de estas variables. Es necesario conocer la influencia que los cambios en los parámetros de la red tienen sobre la función de coste, para esto se debe primero observar el cambio en el error a causa de la salida de la red, para luego propagar esta información hacia las capas anteriores. A este proceso se le conoce como la *retropropagación* o *propagación hacia atrás.*

Así que será necesario recurrir a la célebre regla de la cadena, para encontrar las derivadas parciales necesarias en las ecuaciones 4.14 y 4.15 y así deducir lo que en Aprendizaje profundo se conoce como *Ecuaciones fundamentales de la Retropropagación.*

Para comenzar hay que deducir el cambio en la función de costo respecto a la suma ponderada de la última capa, usando regla de la cadena se obtiene:

$$\frac{\partial C}{\partial z_j^L} = \frac{\partial C}{\partial a_j^L} \frac{\partial a_j^L}{\partial z_j^L},$$

$$\frac{\partial C}{\partial a_j^L} = \left(a_j^L - Y_j \right), \qquad \frac{\partial a_j^L}{\partial z_j^L} = \sigma'(z_j^L).$$

Así, tomando estas componentes se puede calcular $\frac{\partial C}{\partial z^L}$, que no es más que:

$$\frac{\partial C}{\partial z^L} = \Big(a^L - Y\Big) \odot \sigma'(z^L) = \delta^L. \tag{4.16}$$

Donde $\odot$ representa el producto componente a componente entre dos vectores de igual tamaño. La expresión en la Ecuación 4.16 es denotada por δ^L y suele denominarse el error imputado a la capa L. Lo siguiente es analizar cómo cambia la función de coste respecto a las variaciones de los parámetros W^L, b^L en la última capa.

$$\frac{\partial C}{\partial W^L} = \underbrace{\frac{\partial C}{\partial a^L}\frac{\partial a^L}{\partial z^L}}_{\delta^L}\frac{\partial z^L}{\partial W^L}, \qquad \frac{\partial C}{\partial b^L} = \underbrace{\frac{\partial C}{\partial a^L}\frac{\partial a^L}{\partial z^L}}_{\delta^L}\frac{\partial z^L}{\partial b^L}.$$

Nótese que en ambas ecuaciones, los primeros dos factores equivalen al error imputado δ^L, además:

$$\frac{\partial z^L}{\partial W^L} = {a^{L-1}}^T, \qquad \frac{\partial z^L}{\partial b^L} = 1.$$

Por lo tanto al sustituir estos valores, las derivadas parciales quedan como:

$$\frac{\partial C}{\partial W^L} = {a^{L-1}}^T\delta^L, \qquad \frac{\partial C}{\partial b^L} = \delta^L.$$

Y estas dos ecuaciones son las derivadas parciales que necesita el descenso del gradiente para actualizar W^L y b^L, pero esto sólo es para la última capa, entonces es necesario encontrar una manera de deducir el cambio en C debido a las capas anteriores. Esto es relativamente sencillo, basta encontrar el cambio respecto a la capa $L-1$ para encontrar el de todas las capas previas. Para $L-1$:

$$\frac{\partial C}{\partial W^{L-1}} = \underbrace{\frac{\partial C}{\partial a^L}\frac{\partial a^L}{\partial z^L}}_{\delta^L}\underbrace{\frac{\partial z^L}{\partial a^{L-1}}}_{{W^L}^T}\underbrace{\frac{\partial a^{L-1}}{\partial z^{L-1}}}_{\sigma'(z^{L-1})}\underbrace{\frac{\partial z^{L-1}}{\partial W^{L-1}}}_{{a^{L-2}}^T},$$

$$\frac{\partial C}{\partial b^{L-1}} = \underbrace{\frac{\partial C}{\partial a^L}\frac{\partial a^L}{\partial z^L}}_{\delta^L}\underbrace{\frac{\partial z^L}{\partial a^{L-1}}}_{{W^L}^T}\underbrace{\frac{\partial a^{L-1}}{\partial z^{L-1}}}_{\sigma'(z^{L-1})}\underbrace{\frac{\partial z^{L-1}}{\partial b^{L-1}}}_{1}.$$

Sin embargo, los primeros cuatro factores de ambas ecuaciones corresponden a la

derivada

$$\delta^{L-1} = \frac{\partial C}{\partial z^{L-1}} = \frac{\partial C}{\partial a^L}\frac{\partial a^L}{\partial z^L}\frac{\partial z^L}{\partial a^{L-1}}\frac{\partial a^{L-1}}{\partial z^{L-1}},$$

que no es otra cosa que el error imputado de la capa $L-1$. Pero estas derivadas parciales ya han sido deducidas y se conoce su valor, por lo que el error imputado queda como:

$$\delta^{L-1} = \delta^L {W^L}^T \odot \sigma'\left(z^{L-1}\right).$$

Aunque esta expresión se dedujo para el error imputado de la capa $L-1$, es fácil observar (y se puede calcular si se quiere) que, para encontrar el error para la capa $L-2$, se obtendría la misma relación entre esta y la capa que la precede, por lo que esta relación puede generalizarse para cualquier capa $l-1$ además, haciendo un cambio de índices, se obtiene la expresión:

$$\delta^l = \delta^{l+1}{W^{l+1}}^T \odot \sigma'\left(z^l\right). \tag{4.17}$$

Esta misma generalización y arreglo de índice es válida para las expresiones de $\frac{\partial C}{\partial W^{L-1}}$ y $\frac{\partial C}{\partial b^{L-1}}$, de donde se obtienen las expresiones para las derivadas de las capas ocultas

$$\frac{\partial C}{\partial W^l} = {a^{l-1}}^T \delta^l, \tag{4.18}$$

$$\frac{\partial C}{\partial b^l} = \delta^l. \tag{4.19}$$

Finalmente, las ecuaciones 4.16, 4.17, 4.18 y 4.19 se conocen como las *Ecuaciones fundamentales de la retropropagación*, pues con ellas es posible actualizar los parámetros en cada capa de la red aplicando el descenso del gradiente mencionado en las Ecuaciones 4.14 y 4.15, pues al sustituir sus expresiones en ellas se obtiene:

$$W^l \longrightarrow W^l - \eta \nabla_{W^l} C = \eta \sum_x {a_x^{l-1}}^T \delta_x^l, \tag{4.20}$$

$$b^l \longrightarrow b^l - \eta \nabla_{b^l} C = \eta \sum_x \delta_x^l. \tag{4.21}$$

El subíndice x hace referencia a los elementos pertenecientes al conjunto de datos X con el que se esté trabajando, con $x \in X$. La retropropagación es un proceso crucial para el aprendizaje en una red neuronal, ya que los parámetros se actualizan de manera automática siguiendo este algoritmo. Aquí se trabajó con el caso particular del error cuadrático medio; pero estas expresiones son aplicables a cualquier función de coste, pues basta calcular el error imputado correspondiente a la función que se elija.

Para implementar este proceso en una red neuronal, son necesarios los siguientes pasos:

1. Generar valores aleatorios para W^l y b^l, generalmente entre 0 y 1 y siguiendo una distribución normal. Luego calcular la salida correspondiente a^L mediante la propagación hacia adelante.

2. Calcular el valor de la función de coste, la ecuación (4.13), por ejemplo. Luego obtener el error imputado δ^L con respecto a la salida de la red neuronal (Ecuación 4.16).

3. Propagar el error para las capas más profundas, esto es, calcular el error imputado para las capas $L-1, L-2...$ usando la Ecuación (4.17).

4. Utilizar las ecuaciones 4.20 y 4.21 para actualizar los parámetros de la red.

5. Una vez que los nuevos parámetros estén en su lugar, repetir este procedimiento durante un número adecuado de épocas hasta que la función de costo alcance un valor muy pequeño.

Algoritmo 2: Proceso de aprendizaje.

Datos: X, Y.

Paso 1: generar pesos aleatorios W^1 y sesgos b^1 para la red.

para *i in range*(épocas) **hacer**

Paso 2: generar una salida de la red aplicando la propagación hacia delante sobre el conjunto X, luego evaluar la función de coste C con esa predicción y los valores esperados Y.

Paso 3: Actualizar los parámetros de la red con ayuda de las ecuaciones de la retropropagación:

$$W^l \longrightarrow W^l - \eta \nabla_{W^l} C,$$
$$b^l \longrightarrow b^l - \eta \nabla_{b^l} C.$$

fin

5 | Consideraciones prácticas

5.1 Desempeño del modelo: subajuste y sobreajuste

El aprendizaje automático tiene como tarea fundamental dilucidar los patrones que siguen determinados conjuntos de datos. Cuando se trabaja con datos reales se espera conseguir un modelo que describa el fenómeno del que provienen y más aún, poder hacer predicciones con dicho modelo. Para lograr eso, es necesario que este describa no sólo los datos que le fueron provistos para su aprendizaje, también debe hacerlo con datos con los que no ha tenido contacto. Por ejemplo: si se enseña a una red neuronal a clasificar estrellas y galaxias, si la red no puede hacer distinciones entre estrellas y galaxias con las que no fue entrenada, el modelo resultante sería de poca utilidad.

Es de notar que en la práctica es imposible acceder al total de datos que representan una población; únicamente se podrá contar con una muestra, por cuantiosa que esta sea. Esto supone un problema porque cuando un patrón es detectado, este podría ser particular de la muestra, no de la población. Contrariamente, siempre se procurará que las relaciones encontradas en la muestra sean generalizables al total de la población, entonces el modelo se considerará exitoso.

Estas cuestiones dentro del aprendizaje automático han sido estudiadas de manera rigurosa en la rama de las matemáticas que se conoce como *Statistical learning theory* o *Teoría del aprendizaje estadístico*, que no será profundizada en este trabajo, pero algunos de los conceptos que se mencionarán aquí pueden encontrarse con más detalle

en: [41, 42] y [43, p. 142]. En términos simples, en el aprendizaje automático se asume que los datos con los que se trabaja provienen de una determinada distribución P. Así, al abordar el problema de generalización es útil hacerlo a través del error o coste, mencionado en las secciones anteriores. Es posible asumir como *error de entrenamiento* al resultante de evaluar el modelo en la muestra de los datos con la que se cuenta y *error generalizado* al correspondiente a los datos adicionales tomados de la distribución original de la muestra. Como se podría suponer, es virtualmente imposible contar con el error generalizado, pues se trata de una población hipotética y generalmente infinita. Regresando al ejemplo de las estrellas y galaxias, para conocer el error generalizado al clasificar estos objetos, ¡el modelo debería ser evaluado sobre toda la población de estrellas y galaxias del universo!

Así es que, el error generalizado se suele estimar evaluando el modelo en un conjunto proveniente de la distribución original; pero ajeno al proceso de aprendizaje de la red, usualmente denominado **conjunto de validación**. Por lo que se debe contar con dos conjuntos: el *conjunto de entrenamiento*, que representa el desempeño del modelo sobre la muestra y el de validación, que representará la capacidad del modelo de generalizar lo aprendido.

En el aprendizaje automático suelen hacerse ciertas suposiciones sobre los conjuntos de entrenamiento y validación. Por ejemplo: que ambos conjuntos son representativos de su distribución de origen o la famosa hipótesis $i.i.d$, la cual indica que los elementos de entrenamiento y validación son extraídos de la misma distribución de forma independiente [43]. En algunos ejercicios será necesario hacer este tipo de suposiciones; en otros, los supuestos pueden relajarse un poco.

En la sección 3.5 se modeló un conjunto de datos sobre la ley de Hubble con un perceptrón, la red neuronal más sencilla, donde fue posible representar su desempeño mediante su evolución durante el entrenamiento (Fig. 3.5). Generalmente esto no será posible, por ejemplo para el caso en que el modelo dependa de más de dos variables. Por lo que se recurre al comportamiento de la función escalar de coste, analizando su cambio a través de las épocas. Esto es, evaluándola en cada época y graficando el resultado de todo el proceso, lo que permitirá también visualizar el comportamiento

tanto del error de entrenamiento como del de validación.

Expuesto lo anterior, es importante mencionar que existen dos categorías recurrentes a la hora de analizar el desempeño del modelo con ayuda del error. El primero ocurre cuando a través de las épocas, el modelo no reduce el error de entrenamiento. Esto sugiere que el modelo es demasiado simple, por lo que no está asimilando los patrones que subyacen en el conjunto de entrenamiento. A este fenómeno se le llama *underfitting* o infraajuste (Figuras 5.1a, 5.1b) y es posible solucionarlo con un modelo más complejo.

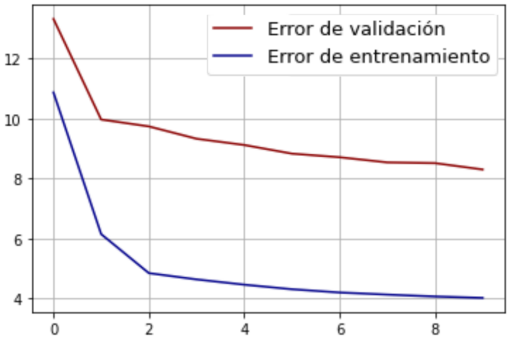
Por otro lado, está la situación en que el error de entrenamiento se encuentra remarcadamente por debajo del error de validación, o más aún: que el error de validación no disminuye a través de las épocas mientras el de entrenamiento sí lo hace. Esta situación es señal de que el modelo es incapaz de generalizar lo aprendido. A este comportamiento se le denomina *overfitting* o sobreajuste (Figuras 5.1c, 5.1d). El sobreajuste puede solucionarse con algunos métodos llamados *regularización*, un par bastante popular es:

1. **Dropout**: Se trata de un procedimiento en el cual se 'apagan' de manera intermitente algunas de las neuronas en cada capa de la red durante el proceso de entrenamiento. Esto significa que a sus respectivas salidas, se les asignará un peso con valor de cero. La elección de a qué neuronas se les aplicará el dropout se realiza de forma aleatoria, siguiendo una determinada distribución de media cero $N(0, \sigma^2)$ [44].

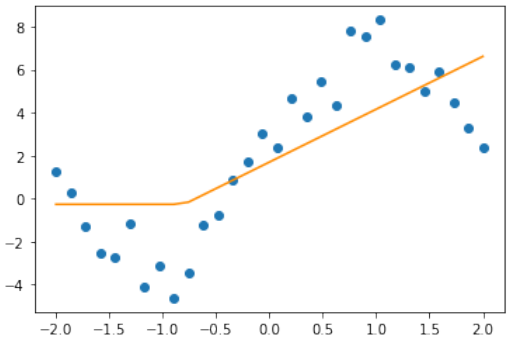
2. **Normas de regularización**: Este método consiste en penalizar a los modelos cuya robustez es grande, es decir aquellos pesos que tengan magnitudes más grandes. Y se definen por ejemplo, extensiones de las funciones de coste como la siguiente:

$$L = C + \lambda \|W^i\|^2.$$

   Donde λ es una constante y L será la nueva norma, que tiene como objetivo regular la magnitud de los pesos asociados y reducir el sobreajuste y C es la función de coste habitual, por ejemplo el error cuadrático medio [45, 46].

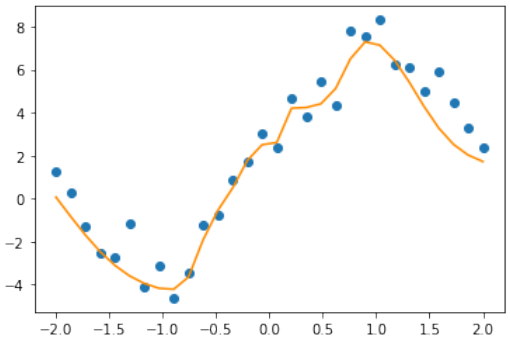


(a) El error de entrenamiento se estabiliza cerca de 4, no disminuye más y se mantiene alejado del error de validación.

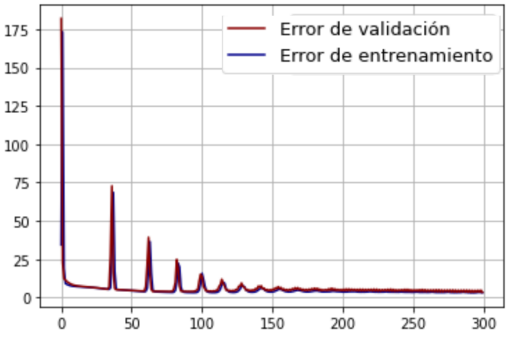


(b) El modelo (línea amarilla) es demasiado simple y, por tanto, no emula eficazmente el conjunto de entrenamiento (puntos azules).

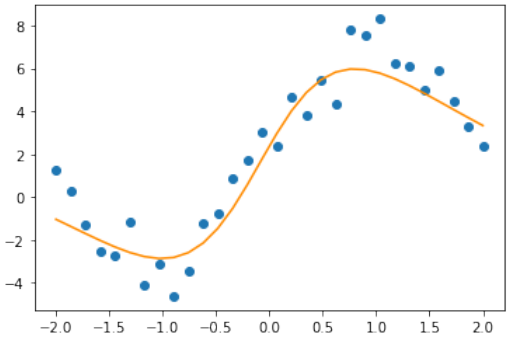
(c) El modelo fue entrenado durante demasiadas épocas, mientras que el error de entrenamiento converge a cero, el error de validación oscila.



(d) El modelo se ha adaptado demasiado bien al conjunto de entrenamiento y es incapaz de generalizar.



(e) La diferencia entre la curvas de entrenamiento y validación es mínima, ambas convergen a cero y el número de épocas es suficiente para un buen ajuste.



(f) Un buen modelo no se rinde ante los datos de entrenamiento, los emula bien y puede generalizarse a más muestras de la distribución.

Figura 5.1: Comparación entre diferentes procesos de entrenamiento y cómo se puede inferir información sobre el rendimiento del modelo mediante la gráfica del error de entrenamiento y validación frente a las épocas.

Por otro lado, si la curva de error como la de validación se mueven a la par y ambas disminuyen a través de las épocas, convergiendo a cero de forma uniforme: esto es síntoma de un proceso de aprendizaje adecuado (Figuras 5.1e, 5.1f).

Finalmente, la práctica es necesaria para aprender a superar estas situaciones, pues tanto el infrajuste como sobreajuste pueden tomar distintas facetas y las curvas de error podrían ser difíciles de interpretar al presentar variedad de comportamientos diferentes a los expuestos aquí. Además, pueden verse potenciadas por factores externos al modelo, tales como: el ruido en los datos, la cantidad de ejemplos que contiene cada conjunto y las épocas, cuya norma general es que un número muy grande de estas termine por sobreajustar el modelo. Como última anotación, es conveniente trabajar con tres conjuntos en el proceso de aprendizaje: El conjunto de entrenamiento, el de validación y un conjunto de prueba, que será en última instancia el que permitirá elegir entre un modelo u otro.

5.2 Preprocesamiento de datos

En la práctica cuando se aborda un fenómeno a través de sus datos, es muy probable que éstos no estén listos para ser procesados por una red neuronal o que necesiten ser tratados para su óptimo manejo. Esto puede deberse a muchos factores, por ejemplo: La presencia de valores atípicos (*outliers*), que la presencia de ruido externo sea importante, una pronunciada diferencia entre el rango de los datos, etc. Se denomina preprocesamiento de datos a la etapa que se encarga de solucionar muchos de estos problemas, como el tratamiento de valores atípicos o transformación y reducción de datos son aplicados aquí. Esta práctica es altamente recomendable para permitir o facilitar el tratamiento de esa información. Además será importante para mejorar el desempeño no sólo de una red neuronal, sino de cualquier modelo.

5.2.1 Datos nulos

En la práctica es común encontrarse con conjuntos de datos incompletos. Ya sea que para ciertos elementos no se aplican determinadas características (las variables inde-

pendientes del conjunto de datos), que no sea posible acceder a esa medición, por un descuido en la captura de los datos o simplemente como consecuencia de un muestreo ineficiente. La ausencia de características en los datos dificulta el tratamiento del fenómeno de estudio, pues limita la información que se tiene acerca de él; además puede llevar a hacer conclusiones erróneas o sesgadas. Existen varios métodos para tratar la ausencia de características o datos nulos y será responsabilidad de la persona encargada del modelado el escoger la que considere más adecuada. Dos de las más recomendadas son [47]:

1. Sustituir el dato atípico por alguno dentro de la muestra, entre las opciones a escoger para esto se encuentran: La moda, la media o la mediana. Hay que tener en cuenta que esta acción también implica inyectar un sesgo a la muestra que no estaba presente en un inicio.

2. Es posible realizar una regresión o clasificación sobre el conjunto de datos, tomando la etiqueta del dato faltante y poniéndolo en función de los otros. Al hacer esto, la relación resultante se asume como válida y se predice un valor que sustituya al faltante. Curiosamente las redes neuronales son especialmente buenas para realizar esta tarea.

5.2.2 Tratamiento de datos atípicos

Los valores atípicos en una muestra aleatoria son aquellos que podríamos describir de forma simplificada como observaciones cuyo valor está muy alejado del resto de los datos. Su detección y tratamiento es muy importante, pues pueden tener mucha influencia en la muestra que estamos considerando y por consecuencia afectar nuestra descripción, suposiciones o predicciones sobre la población en general.

La primer y más sencilla manera de detectar este tipo de elementos, es mediante la visualización de la distribución de cada una de las variables que estemos considerando. Una manera sencilla de visualizar dicha distribución es mediante el diagrama de caja, un arreglo que pone la muestra como una caja rectangular dividida en cuartiles, que representan el 25 % y 75 % de las observaciones de una muestra, luego, en los

extremos encontramos los llamados brazos o bigotes, que alcanzan un máximo y un mínimo dentro de la distribución. Lo más importante es que las observaciones que se encuentren fuera de los bigotes son valores atípicos. Existen más maneras de identificar a los outliers, ya sea mediante métodos estadísticos, como la normalización de punto z que se desarrolla más adelante, o con ayuda de algoritmos computacionales, el cuál elegir dependerá del contexto del problema.

Una vez detectados los valores atípicos, se pueden tratar de distintas maneras, entre las más comunes estarían:

1. Imputar el o los elementos, esta puede ser una opción viable para muestras con muchos elementos, pero es poco recomendable debido a que existe una pérdida de información de la muestra, que puede ser muy relevante para la descripción de la población y su descripción.

2. Una opción mucho mejor es eliminar el valor atípico para luego sustituirlo por uno más conveniente. Al eliminarlo, se tiene la situación de dato faltante y se pueden aplicar las soluciones ya discutidas a detalle en la sección anterior.

3. También es posible mantener el dato atípico, entendiendo que este legítimamente forma parte de la población, pero bajo la responsabilidad que pueda influir en los resultados obtenidos a partir de la muestra.

5.2.3 Transformación de los datos

Es importante que los datos con los que se trabaja estén dentro de un rango de valores similares. En especial cuando el objetivo es modelarlos con una red neuronal: puesto que, como se mencionó en la sección 3.2, los pesos representan la importancia de la entrada correspondiente en la neurona. Si una de las entradas x_i se encuentra en un rango numérico de orden superior a las otras, tendrá una mayor contribución a la suma ponderada de su capa, por lo que su influencia sobre la salida de la red se verá pronunciada. La red neuronal aprenderá esto y asignará una mayor importancia en el modelo a x_i a través de su peso asociado y sus conexiones siguientes, aunque

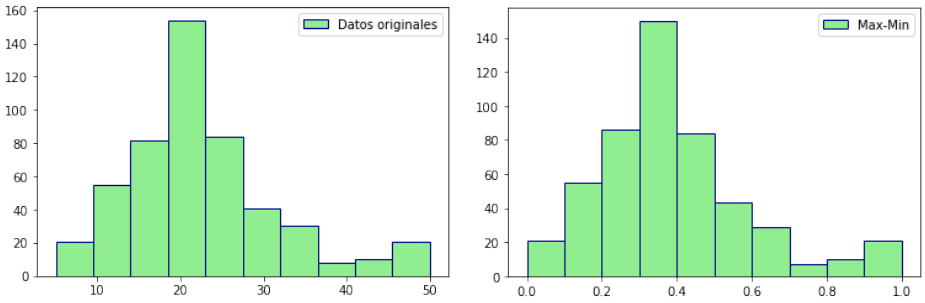
dicha importancia fue impuesta de manera artificial. Para solucionar este problema se suelen aplicar transformaciones a las distribuciones de los datos [47], en palabras sencillas se busca conservar su estructura pero modificando su escala. Dos de las transformaciones más comunes son:

- **Transformación por máximos y mínimos**

  Consiste en tomar los datos originales X con sus respectivos máximos y mínimos para mapearlos en un intervalo X' con un nuevo máximo y mínimo. Esto se consigue haciendo la transformación

$$X' = \frac{X - \min(X)}{\max(X) - \min(X)}(\max' - \min') + \min', \tag{5.1}$$

  donde $\max'$ y $\min'$ representan los nuevos máximo y mínimo respectivamente. Como se muestra en la figura 5.2, la forma de la distribución no se altera, pero el conjunto transformado ahora se encuentra en el nuevo intervalo que se le fue asignado.



(a) Datos originales en un intervalo [0,50].

(b) Transformación Max-Min, donde los extremos del nuevo intervalo son 0 y 1.

Figura 5.2: Transformación por máximos y mínimos aplicada a un conjunto de datos arbitrario.

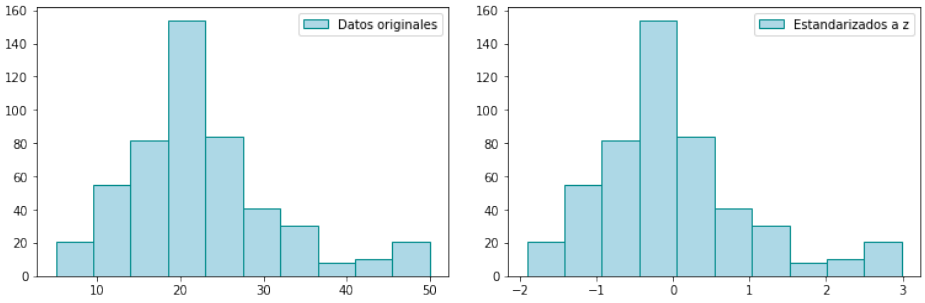
- **Normalización de punto z**

  Esta transformación no debe interpretarse como que los datos nuevos seguirán una distribución normal, aunque el nombre pudiera sugerirlo. Se trata de un proceso que aplica una transformación en la distribución datos, de tal suerte

que la distribución resultante cumple con: estar centrada en cero, esto es su media vale cero y tener una varianza igual a uno, $\sigma = 1$. Dicha transformación se define como:

$$X' = \frac{X - \bar{X}}{\sigma}, \tag{5.2}$$

con σ, $\bar{X}$ la desviación estándar y la media aritmética de la distribución original respectivamente. De nuevo, esto se puede apreciar en la Figura 5.3, pues al transformar los datos ahora el promedio se encuentra en 0 y si se calcula su varianza, esta resultará en 1.



(a) Datos originales

(b) Normalización a z: con la distribución ahora centrada cen cero y con varianza de 1.

Figura 5.3: Transformación de punto z.

Estas transformaciones suelen ser aplicadas al conjunto de entradas de la red, dejando a las etiquetas o valores esperados Y como estaban en un inicio. Sin embargo, cuando la diferencia en el rango de los elementos de Y es muy pronunciado es buena idea aplicar alguna transformación también a este conjunto. Es menester recalcar que una vez que se ha definido la transformación a los datos, esta misma deberá ser aplicada en todas las ocasiones posteriores, para que exista una congruencia en la transformación, por ejemplo: si se han utilizado la media y desviación estándar del conjunto de entrenamiento para hacer una normalización de punto z, esta misma deberá utilizarse para el conjunto de validación y el de prueba, así como para cualquier futura predicción. De igual manera, si se aplica una transformación sobre el conjunto Y, y se requiere que el valor de la predicción de la red esté en términos de la distribución original, se deberá aplicar la transformación inversa a la salida de

la red.

5.2.4 Selección de características

En las bases de datos utilizadas en el aprendizaje automático, se debe identificar si existen características del conjunto que son irrelevantes o que puedan aportar información redundante: por ejemplo en la clasificación de objetos, si alguna de las características es común para todas las clases, en consecuencia, su presencia es irrelevante para que el modelo aprenda. Al proceso de reducir el espacio de atributos a uno menos redundante se le conoce como *selección de características*. Algunas de los métodos propuestos por [48] para esta tarea son:

- El análisis de componentes principales (PCA) se trata de un método aplicable principalmente a matrices con un número pequeño de características. Se caracteriza por encontrar un cambio de coordenadas tal que el conjunto de características posiblemente correlacionadas se transforme en un conjunto de características linealmente no correlacionado [49].

- Selector Chi-Cuadrada: este método para características categóricas utiliza una prueba Chi-Cuadrada de independencia de la clase, para seleccionar un número preestablecido de las variables que muestren ser más dependientes.

- Un método no tan riguroso pero igualmente aplicable es el denominado *vector slicer*. Aquí el sujeto selecciona las características que formarán parte del conjunto de características reducido de forma manual, basándose en un análisis heurístico de los datos.

Estas son algunas de las maneras de reducir el conjunto de características, que permitirán no sólo disminuir la dimensionalidad del conjunto con el que se trabaja, sino que mejorará el desempeño del modelo y optimizará el coste computacional. Sin embargo, este tema es complejo y extenso, por lo que un análisis más a fondo sobrepasa el objetivo de este trabajo, así que se recomienda consultar literatura especializada en el tema para una mejor comprensión, se sugiere [50].

5.3 Red neuronal desde cero

Aquí se discutirán algunos aspectos finales sobre la implementación y el entrenamiento de un MLP desde cero. Para los detalles técnicos, el código completo relacionado tanto con esta sección como con el resto, se puede encontrar en [51].

Antes de todo, los conjuntos de datos que se utilizarán durante el proceso de aprendizaje deben ser preparados como se comentó en las secciones (5.1, 5.2). Una vez hecho eso, se debe definir la arquitectura del perceptrón multicapa (MLP); es decir, el número de capas y su respectivo número de nodos, así como la función de activación que corresponde a cada capa. Se dice en aprendizaje profundo que la elección de la arquitectura y los hiper-parámetros de la red es más un arte que una ciencia, obviando el hecho que entre mayor sean el número de capas ocultas y el número de nodos, el modelo será más complejo.

La propagación hacia delante de una red neuronal es un proceso vectorizado, lo que significa que procesa todo el conjunto de entradas (generalmente un arreglo matricial o tensor) en una sola pasada. Así que la capa de entrada debe estar determinada por el número de variables independientes del conjunto de características. Sin embargo, si existen demasiados elementos en el conjunto de entrenamiento, se suele tomar pequeños lotes (subconjuntos más pequeños del original), y procesarlos en la red en vez del total, con el fin de reducir el coste computacional. Cuando se intenta optimizar la función de coste tomando lotes (batch) en vez del conjunto total, se dice que el descenso del gradiente se vuelve estocástico y el número de elementos en cada lote se denomina *batch-size*.

Una vez que la arquitectura y el conjunto de datos están listos, se establecen unos valores aleatorios para los pesos W^l y los sesgos b^l, generalmente con componentes entre 0 y 1, por lo general siguiendo una distribución uniforme o una normal centrada en ese intervalo. Entonces se genera una primera predicción del MLP y el algoritmo de retropropagación comienza a actuar. Ajustando los parámetros del MLP, entonces las predicciones se acercan más a los datos reales, y en consecuencia, este proceso debe tener tantas épocas como sean necesarias para minimizar la función de coste y obtener

un modelo que describa adecuadamente los conjuntos de datos de entrenamiento y validación. Si el modelo muestra un buen aprendizaje y si es capaz de generalizar al conjunto de validación, entonces la red neuronal puede guardarse en un archivo binario. De este modo, los valores de las matrices de pesos W^l y los vectores de sesgo b^l obtenidos tras un entrenamiento adecuado se guardan para realizar predicciones, sin necesidad de entrenar la red neuronal una vez más.

Construir una red neuronal desde cero demuestra que se han entendido los conceptos fundamentales del Deep Learning. Dicho esto, en la práctica es necesario trabajar con diferentes arquitecturas de red, funciones de coste y activaciones; por lo que construir la red desde cero cada vez puede ser una tarea difícil, dado que al cambiar de arquitectura deberá replantearse tanto la propagación hacia delante como el algoritmo de retropropagación. Más aún, en la práctica generalmente se implementan varias arquitecturas de red para un mismo problema, pues encontrar la que dé mejores resultados es más un arte que una ciencia. Así mismo respecto a las tasa de aprendizaje y las funciones de activación. Es por ello que existen diferentes librerías especializadas en Deep Learning en Python como: Pytorch, TensorFlow o Keras. Cuya esencia radica en computar los cambios (derivadas parciales) en la función de coste respecto de cada parámetro de la red de forma automática, por lo que resultan ser bastante prácticas al automatizar el montaje de una red neuronal, especialmente lo que corresponde al proceso de retropropagación. En este trabajo se realizará el desarrollo e implementación de una red neuronal desde cero (Sección 6.1) y dos con Keras (Secciones 6.2, 6.3).

6 | Aplicaciones cosmológicas

En este capítulo se aplicará todo lo desarrollado sobre redes neuronales para abordar tres problemas de investigación concernientes a la cosmología. El primero es implementar una red neuronal construida desde cero como se mencionó en la sección 5.3, y utilizarla para generar un par de modelos para dos conjuntos de datos diferentes: uno que contenga una serie de datos que sigan la ecuación de Hubble con una incertidumbre asociada, el segundo será un conjunto de valores correspondientes a distintas variaciones en los parámetros de la ecuación de Friedmann. Así, la red debe mostrar que es capaz de emular tanto el comportamiento de los datos como el de su incertidumbre en función del corrimiento al rojo, o encontrar distintos modelos que describan de forma correcta un conjunto de datos dependiente de muchos parámetros.

En segundo lugar, se abordará el tratamiento de ecuaciones diferenciales de forma numérica, pero aplicando redes neuronales artificiales. Mostrando así que el uso de esta rama de la inteligencia artificial permite acelerar el tiempo de cómputo para obtener las soluciones de un sistema de ecuaciones diferenciales, en este caso uno concerniente a la evolución de los parámetros cosmológicos de densidad. Más aún, permitirá contar con un modelo funcional de las soluciones para cualquier sistema de ecuaciones, algo equivalente a contar con las expresiones analíticas desde el punto de vista computacional.

Finalmente, se abordará la clasificación de objetos astronómicos. Una tarea que puede ser complicada debido al gran número de variables que intervienen, así como la enorme cantidad de datos observacionales que se deben procesar. También cabe mencionar que además del procesamiento de los datos, queda la tarea de interpretación para asignar clases a los objetos. Aquí se entrenará una red neuronal a partir de un

conjunto de datos con objetos ya clasificados dentro de tres clases distintas en función de sus características.

6.1 Modelando la ecuación de Friedmann

Como ya se mencionó, las redes neuronales actúan como un aproximador universal de casi cualquier función. Esta cualidad puede ser explotada de muchas maneras en la ciencia, en especial cuando se desea tener un modelo a partir de ciertos datos observacionales. En este ejemplo, se implementó un perceptrón multicapa desde cero para crear un modelo basado en algunos datos simulados de la teoría cosmológica. Con un par de propósitos: obtener un modelo capaz de generalizar un conjunto de datos y su incertidumbre asociada y un segundo, donde se tendrán datos provenientes ya sea de distintos modelos o de las variaciones en los parámetros de algún único modelo. Ambas redes están disponibles en el repositorio.

Los datos utilizados aquí fueron generados de manera artificial, con el propósito de tener un referente teórico para la comparación e interpretación de la red neuronal resultante. En este caso se utilizó la Ecuación de Friedmann en su forma mostrada por la ecuación 2.25.

6.1.1 Cómo modelar datos y sus incertidumbres

Para el primer problema de investigación, se asumió un Universo con geometría plana, cuya evolución se describe en épocas tardías y la contribución de la radiación es despreciable, afirmación sustentada en que en las mediciones actuales $\Omega_{r,0}$ tiene una contribución aproximadamente 5 órdenes de magnitud inferior a la correspondiente a la materia o la energía oscura. Por otro lado, como los parámetros adimensionales están normalizados, sus valores de la época actual cumplen $\Omega_{\Lambda,0} = 1 - \Omega_{m,0}$. También se establecieron los valores de $\Omega_{m,0} = 0.27$ y $H_0 = 73.24$, estos valores fueron tomados como representación de lo que serían las observaciones reales, pues como se mencionó al final de la sección 2, hay ciertas discrepancias respecto al valor real de

estos parámetros. Una vez mencionado eso, la ecuación 2.25 se convirtió en:

$$\frac{H^2}{H_0^2} = \Omega_{m,0}(1+z)^3 + \Omega_{\Lambda,0}. \tag{6.1}$$

Se generaron 20 datos artificiales a partir de (6.1) y con la ayuda del método dado por la red neuronal publicada por [13]. Adicionalmente se le inyectó un ruido aleatorio tanto a los datos como a sus barras de error asociadas, este ruido se generó sumando a cada valor $H(z)$ y su respectiva barra de error, un número aleatorio $\epsilon, \delta \in [-0.1, 0.1]$ multiplicado por el propio valor, es decir, el ruido se asigna de manera uniforme entre el -10 % y 10 % del valor que corresponda en determinado z:

$$H(z)_{\text{noisy}} = H(z) + \epsilon H(z),$$

$$\text{error}(z)_{\text{noisy}} = \text{error}(z) + \delta\, \text{error}(z).$$

Así que los datos que servirán como entrada para el aprendizaje de la red neuronal, se muestran en la Figura 6.1.

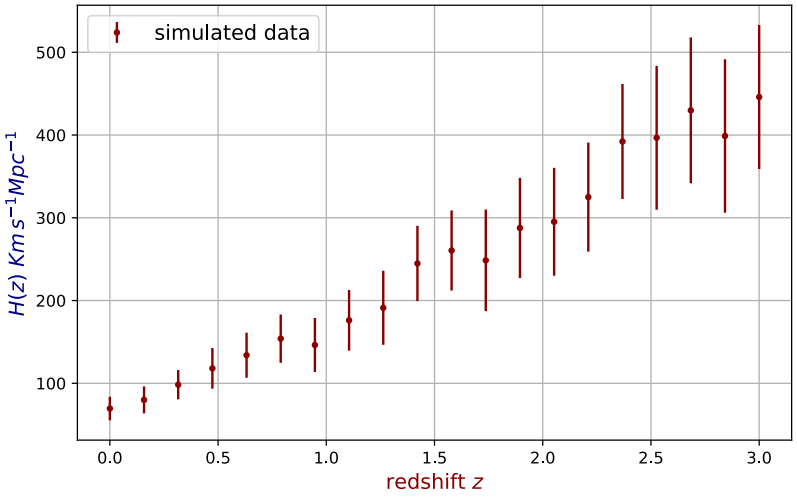


Figura 6.1: Datos simulados para el entrenamiento de la red. Tanto los puntos como las barras de error tienen un ruido aleatorio uniforme asociado, cuyo valor puede ir del -10 % al 10 %.

Tanto el valor de la función como el error fueron dados a la red neuronal en el

papel de las etiquetas, ambas en función del corrimiento al rojo z, proporcionado como entrada. La arquitectura de red utilizada para este ejemplo se muestra en la Figura (6.2).

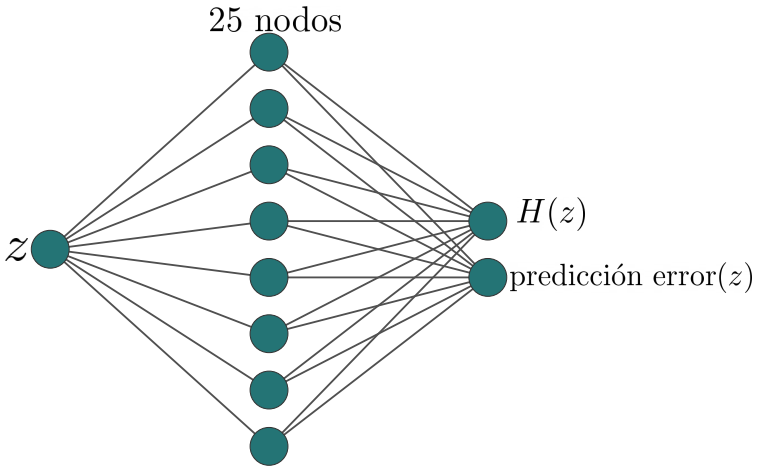


Figura 6.2: Arquitectura de red neuronal implementada para la ecuación de Friedmann. La capa de entrada recibe el corrimiento al rojo y la capa de salida genera un valor para el parámetro de Hubble y su respectivo error.

Una vez que la RNA completó su proceso de entrenamiento, pudo generalizar tanto los datos como sus incertidumbres asociadas sobre todo el dominio z, como se puede apreciar en la Figura (6.3).

6.1.2 Datos provenientes de varios parámetros o modelos

Es posible encontrarse con observaciones de fenómenos que dependen de muchas variables, o cuya descripción teórica sea dependiente de una serie de parámetros y las observaciones recogidas presenten una combinación de todos ellos. Para abordar este problema sería necesario a partir de dichas observaciones contar con uno o varios modelos para verificar cuál es mejor según la teoría correspondiente.

Para tratar este problema a través del aprendizaje profundo, se tomó la ecuación 6.1, que es dependiente de tres parámetros H_0, $\Omega_{m,0}$ y $\Omega_{\Lambda,0}$. Debido a que los parámetros cosmológicos están normalizados, se reduce la dependencia a sólo dos de ellos, de donde se eligió variar aquellos cuya magnitud es de mayor interés y debate: Ω_m y H_0 igual que en la red anterior. El espacio de parámetros se restringió entonces a

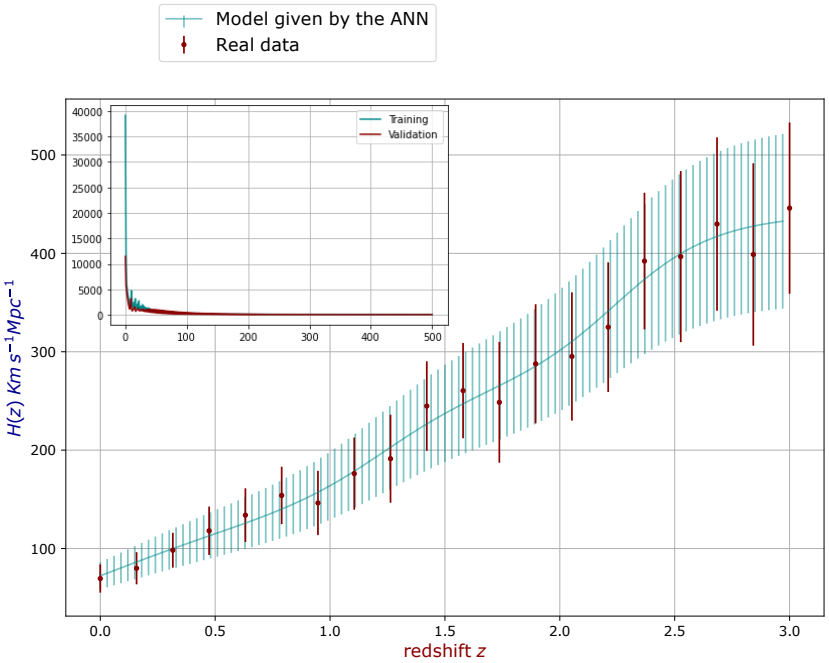


Figura 6.3: Aquí se muestra el modelo entrenado. Las barras de error rojas son los datos originales utilizados para entrenar la red. Las barras de error de color turquesa representan las predicciones de la RNA entrenada. En la esquina superior izquierda se muestra el comportamiento de la función de pérdida en los conjuntos de entrenamiento y validación, que tal y como se explicó en la sección 5.1, tiene un buen comportamiento, pues tienden ambas curvas a cero, poseen valores muy similares y no hay irregularidades a través de las épocas.

los intervalos $\Omega_{m,0} \in [0.2, 0.5]$, $H_0 \in [0.66, 0.70]$ y $z \in [0, 3]$ y en ellos se evaluó en la ecuación 6.1. Esto da como resultado un conjunto de datos que dependen del corrimiento al rojo y donde cada elemento pertenece a una combinación de parámetros particular, como se puede apreciar en la Figura 6.4. Para que la red intuya qué debe aprender de los datos y cómo debe relacionarlos, su conjunto de entrenamiento se generó de tal manera que los datos son la imagen de una función escalar hipotética dependiente de 3 variables:

$$\mathcal{F}(\Omega_{m,0}, H_0, z) \longrightarrow H.$$

Así la red resultante podría ser evaluada en cualquier combinación de esas variables, que de hecho son los parámetros de la ecuación de Friedmann. Entonces, a partir

de esa función, se creó una malla de valores producida por el producto cartesiano de los intervalos correspondientes a cada variables que se mencionaron arriba.Con esto se generó un conjunto de atributos y uno de etiquetas con 4650 datos cada uno. Debido a que su gráfica es de cuatro dimensiones, para tener una idea de cómo se ve el conjunto de datos, se puede visualizar en el plano formado por H y z, como se ve en la Figura 6.4. La arquitectura utilizada para esta tarea se muestra en la Figura

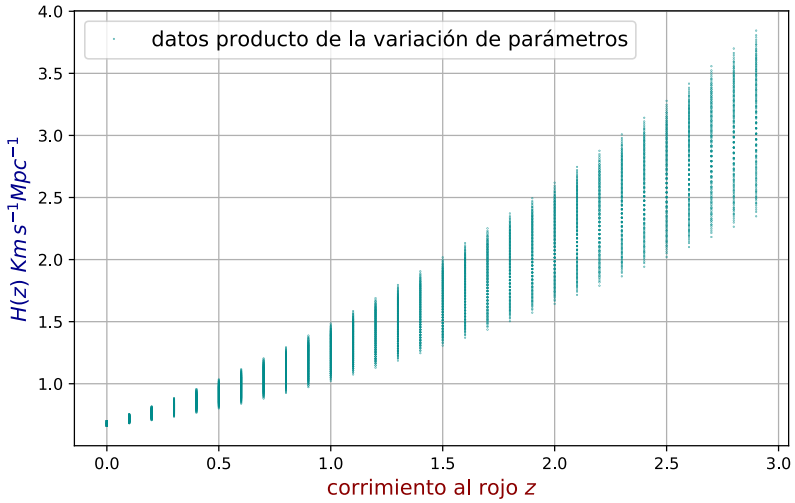


Figura 6.4: Datos a partir de la variación de los parámetros de Friedmann, los puntos que forman líneas verticales corresponden a distintas combinaciones de parámetros de la ecuación de Friedmann evaluados en un mismo z.

(6.5). El MLP se entrenó durante 1000 épocas, donde según sus curvas de error, no se presentó ni sobreajuste ni infrajuste. Finalmente, la red entrenada fue capaz de aprender a emular la función $\mathcal{F}$, en consecuencia pudo proporcionar una colección de modelos para cualquier combinación de parámetros $\Omega_{m,0}$ y H_0 en los respectivos intervalos de donde se generaron los datos de entrenamiento.

En este par de ejemplos, se lograron los objetivos iniciales: evidenciar que una red neuronal puede aprender el comportamiento de cualquier conjunto de datos, en este caso, provenientes de una función determinada. Este hecho es importante pues dado un conjunto de datos experimentales u observables, es posible asignarles un modelo que los describa de buena manera y de bajo costo computacional, pues es una función y no un método numérico. Además de ser fácil de implementar, pues para ambos

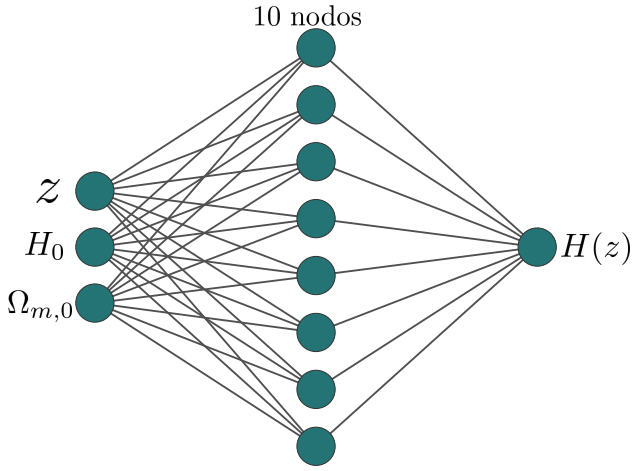


Figura 6.5: Arquitectura de red neuronal para aprender el parámetro de Hubble H, dado un conjunto de parámetros cosmológicos: desplazamiento al rojo z, H_0 y $\Omega_{m,0}$.

ejemplos se usaron redes neuronales muy sencillas, como se puede observar de las arquitecturas 6.2 y 6.5, los componentes de la red son apenas algunos coeficientes numéricos constantes: los pesos y sesgos.

Ambos casos tienen aplicaciones prácticas, tales como: obtener un modelo que se puede generalizar y evaluar cuando sea necesario, completar conjuntos de datos pobres, comparar muchos modelos obtenidos de la variación de los parámetros de una función multidimensional y elegir cuál es el mejor. Como se tiene que la red neuronal es una relación funcional, permite hacerle todo tipo de manipulaciones matemáticas, como: derivar, integrar, obtener puntos críticos, graficar, etc.

6.2 Ecuaciones diferenciales cosmológicas

Como se mencionó en la sección 2.1, a través de las ecuaciones diferenciales 2.27 se puede encontrar cómo han cambiado el universo y su contenido desde su origen. Dichas ecuaciones diferenciales no son difíciles de resolver de forma manual, pero son un buen ejemplo para mostrar cómo se puede optimizar el tiempo de cómputo en la obtención de las soluciones de un sistema de ecuaciones diferenciales. Esto puede verse potenciado cuando es necesario hacer muchísimas evaluaciones para distintas condiciones iniciales o parámetros del sistema. Por ejemplo en las simulaciones compu-

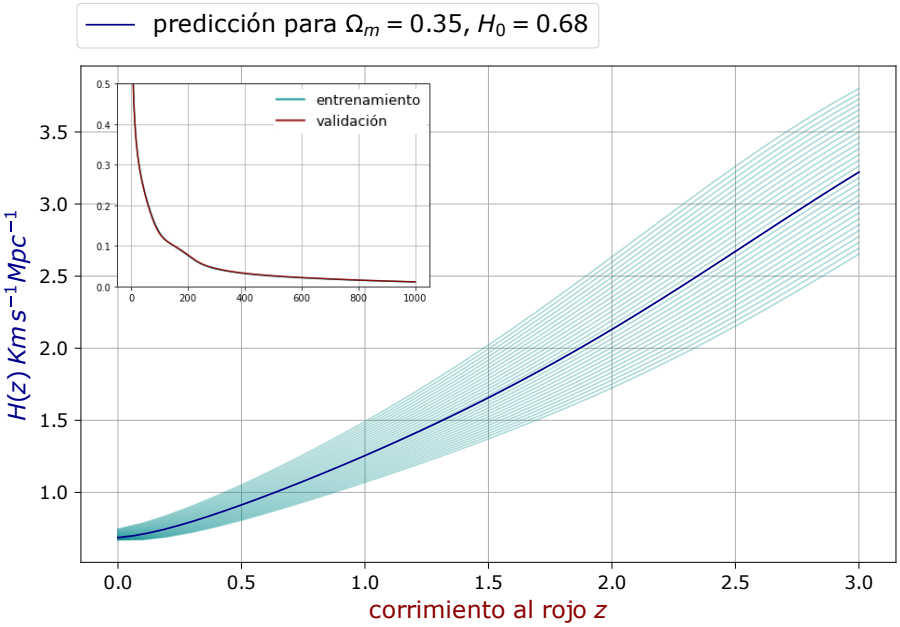


Figura 6.6: Con los datos correspondientes a diferentes parámetros de la ecuación 6.1, la red proporcionó una colección general de modelos. Cada línea azul indica un modelo generado por la red neuronal para una combinación específica de parámetros cosmológicos. La línea azul oscuro corresponde a $\Omega_{m,0} = 0,35$ y $H_0 = 0,68$. En la esquina superior izquierda, se puede observar el comportamiento de la función de pérdida para los conjuntos de entrenamiento y validación.

tacionales o el análisis estadístico del espacio de parámetros con cadenas de Markov Monte Carlo.

Para este problema de investigación, se considerará un universo plano cuya evolución se remonta a tiempos tempranos de la existencia. De igual manera se supondrá que el parámetro de radiación a tiempo presente cuenta con un valor de gran fiabilidad, por lo que se puede fijar en $\Omega_{r,0} = 0.0001$. Una vez que se establece el valor de $\Omega_{r,0}$, ya que la suma de todas las densidades debe ser igual a uno (equivalente al contenido total de un Universo plano), basta con variar solo los parámetros $\Omega_{m,0}$ y H_0 para tener las condiciones iniciales para el sistema 2.27.

Si una ANN se entrena usando soluciones de ecuaciones diferenciales provenientes de diferentes condiciones iniciales como datos de aprendizaje, el modelo resultante podría dar soluciones para cualquier conjunto de condiciones iniciales, sin necesidad de resolver el sistema de ecuaciones nuevamente. Esto se basa en la misma idea que el

ejemplo de la sección 6.1.2, pues aquí se transformará la ecuación diferencial en una función de los parámetros que la definen:

$$\Omega_i'\left(\ln(z)\right) \longrightarrow \mathcal{F}\left(\ln(z),\, \Omega_m,\, \Omega_\Lambda\right). \tag{6.2}$$

Como se explicó al final de la Sección 5.3, armar una red desde cero cada vez que se desee procesar datos puede ser un proceso tedioso y poco práctico. Es por eso que en la práctica se recurre a las librerías especializadas. El perceptrón multicapa utilizado en el ejercicio anterior solo permite una capa oculta, y para cambiar esa arquitectura, sería necesario volver a calcular sus ecuaciones correspondientes al algoritmo de retropropagación. Por este motivo, dado que este problema es más complejo, se ha utilizado una red neuronal implementada con la librería `Keras`.

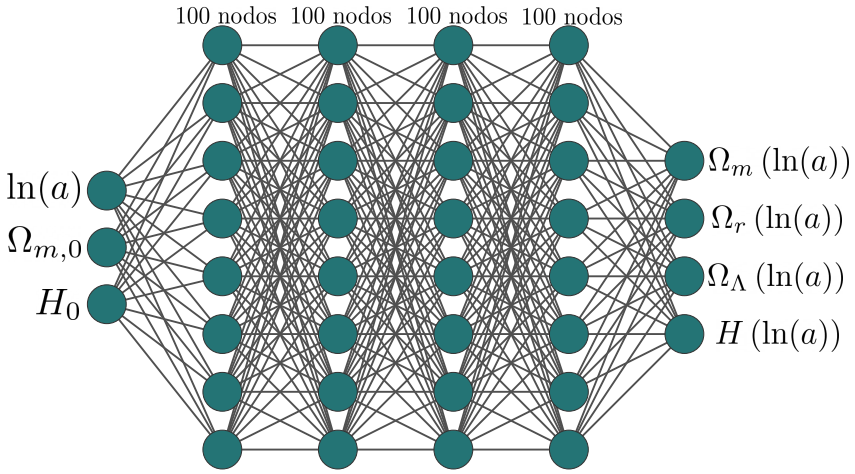


Figura 6.7: Arquitectura de la red utilizada para emular las ecuaciones diferenciales.

El conjunto de datos se generó a partir de la definición de intervalos para las condiciones iniciales: $\Omega_{m,0} \in [0.1, 0.5]$ y $H_0 \in [65, 80]$. Posteriormente, se creó una malla con todas las combinaciones entre ambos intervalos de valores, lo que resultó en un conjunto de datos para el aprendizaje de la red con 30,750 elementos, las respectivas soluciones se obtuvieron a partir de una librería especializada en EDO's como es `odeint` de `python`. A diferencia del ejemplo anterior, donde los atributos se encontraban dentro de un mismo rango (todos entre cero y uno); aquí las soluciones

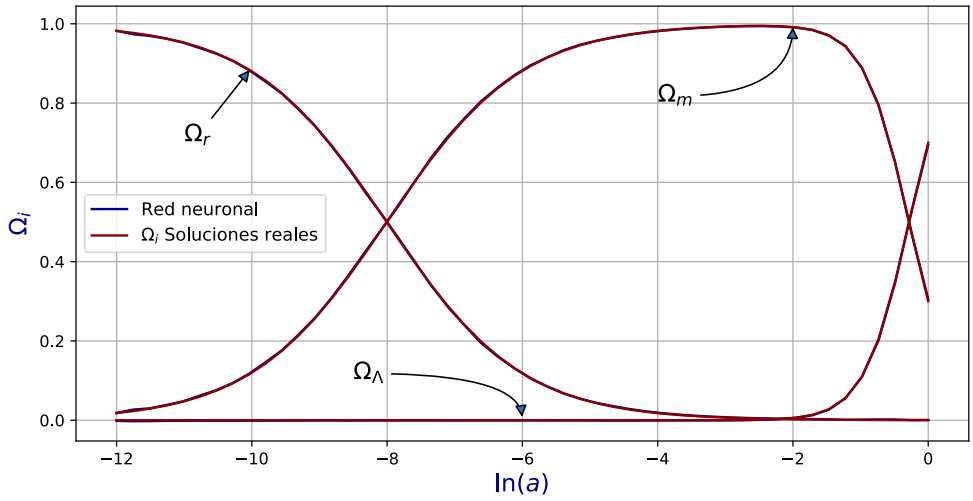
del sistema son los parámetros adimensionales de densidad, que están normalizados y el de Hubble, que para valores muy tempranos en la edad del universo alcanza valores del orden de 10^{10}. Esta enorme diferencia de rangos hace que la solución para H cargue con más influencia en el modelo y evite que las demás soluciones se describan de forma adecuada. Esto se abordó haciendo una transformación de max-min no a los atributos del conjunto de datos, sino a las etiquetas, las soluciones de las ecuaciones.

La red neuronal se entrenó durante 500 épocas, utilizando la arquitectura que se muestra en la Figura 6.7, con activación sigmoide en las capas ocultas. De esta manera, la red neuronal aprendió a generar las soluciones simplemente con evaluar los parámetros de densidad del tiempo actual y el logaritmo del factor de escala que se requiera. En la Figura 6.8 se comparan las soluciones reales y las predicciones hechas por la red para los parámetros $\Omega_{m,0} = 0.3$ y $H_0 = 70$, y su evolución desde los primeros momentos después del Big-Bang.

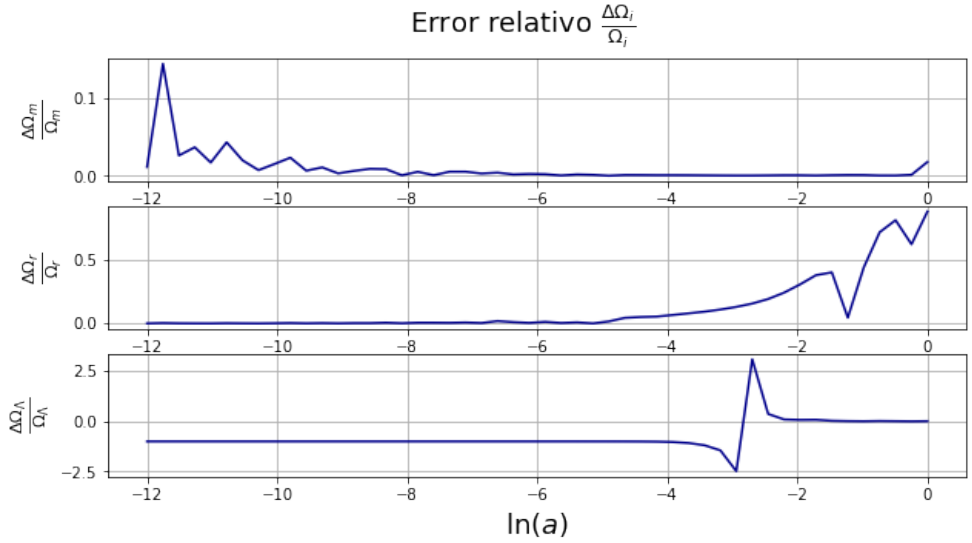
Se puede apreciar el cambio en la densidad de la radiación, la materia y la energía oscura. Donde se concluye que: aunque hoy en día su aportación es nimia, el universo joven estuvo dominado por la radiación. Por otro lado, se especula que cuando la densidad de la radiación fue disminuyendo, se dio paso a las formaciones de nucleones: comenzaron a formarse los elementos ligeros, algo que también está en concordancia con los resultados de este sistema. Finalmente, los cosmólogos asocian a la densidad de energía oscura la propiedad de 'no diluirse', a pesar de la expansión del universo; lo que explica por qué a pesar de tener unas contribuciones muy pequeñas, hoy en día su densidad es la dominante en el cosmos.

Por su diferencia de rango, la solución y comparación para el parámetro de Hubble evaluado con las mismas condiciones iniciales, se presenta en la Figura 6.9. Donde se observa la solución correspondiente emulada de buena manera por la RNA. También se muestran las curvas de error para el entrenamiento de esta red, donde a pesar de haber ciertas fluctuaciones en el error de validación, este junto con el de entrenamiento son muy cercanos a cero.

Una vez entrenada la red, se puede guardar el modelo para evaluarlo siempre que sea requerido. En este caso también se le pidió a la red que entregara las soluciones



(a)



(b)

Figura 6.8: En 6.8a se muestran tanto las soluciones numéricas reales como las obtenidas al evaluar el modelo de la red neuronal entrenada. Ambas con los parámetros $\Omega_{m,0} = 0.3$, $H_0 = 70$, evaluados sobre todo el dominio N. Luego, en 6.8b se muestra el error relativo correspondiente para cada una de las soluciones correspondiente a cada z.

para distintos parámetros iniciales, algunos de los resultados se pueden apreciar en la Figura 6.10. Donde se aprecia que la variación en los parámetros iniciales, si bien no cambian la forma general de las soluciones, sí lo hace numéricamente. Aunque el análisis para cada combinación de condiciones iniciales resulta similar al hecho

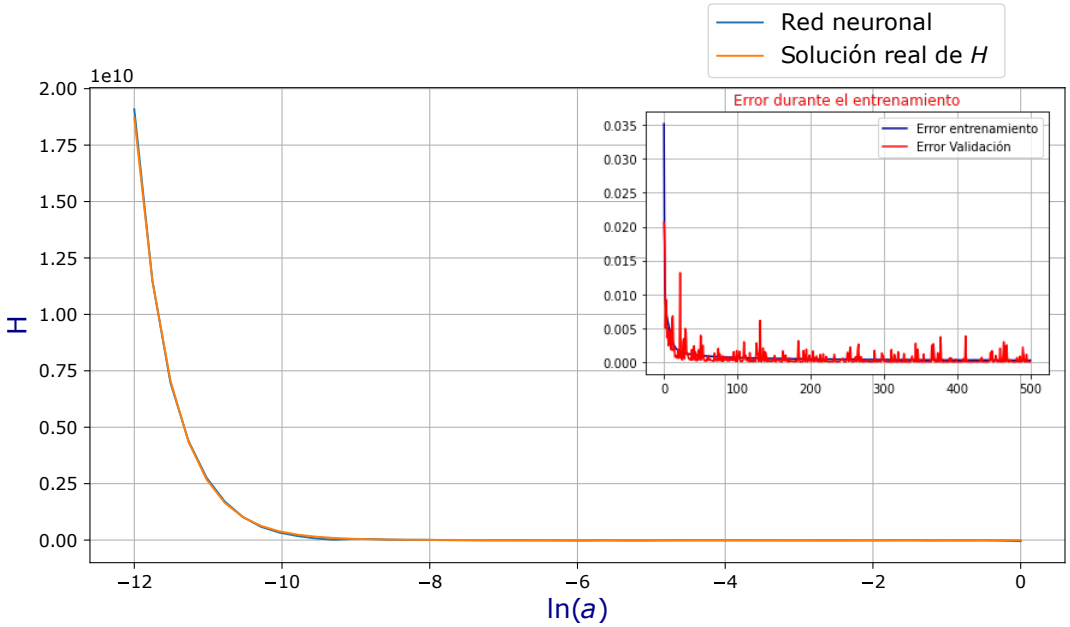


Figura 6.9: Comparación de la solución del parámetro de Hubble real y la dada por la RNA.

para las condiciones iniciales $\Omega_{m,0} = 0.3$ y $H_0 = 70$, la etapa en la que suceden las variaciones sí cambia. Por ejemplo, el paso del dominio de la radiación por el de la materia sucede más en el pasado conforme mayores son $\Omega_{m,0}$ y H_0. A la época en la vida del universo cuando se dieron las condiciones para que iones y los electrones pudieran comenzar a formar átomos, se le conoce como *etapa de recombinación.*

Finalmente, para probar la eficiencia del modelo, se procedió a hacer una prueba de complejidad computacional frente a un método de resolución numérico de una librería especializada en `Python`: `integration.odeint` de la biblioteca llamada `Scipy`. Para la prueba se generó un conjunto de $10,000$ datos para diferentes combinaciones de condiciones iniciales. Con el fin de que la comparación fuera lo más equilibrada posible, se vectorizó el método de solución numérico y las soluciones le fueron requeridas utilizando un mapeo, puesto que los ciclos son bastante ineficientes en comparación, también se tomó la transformación inversa a la hecha en el preprocesamiento y se aplicó a la salida de la red. El resultado fue que, en porcentajes, la red neuronal redujo el tiempo de cómputo para obtener las soluciones a sólo un $48\,\%$ del que tardó el método numérico.

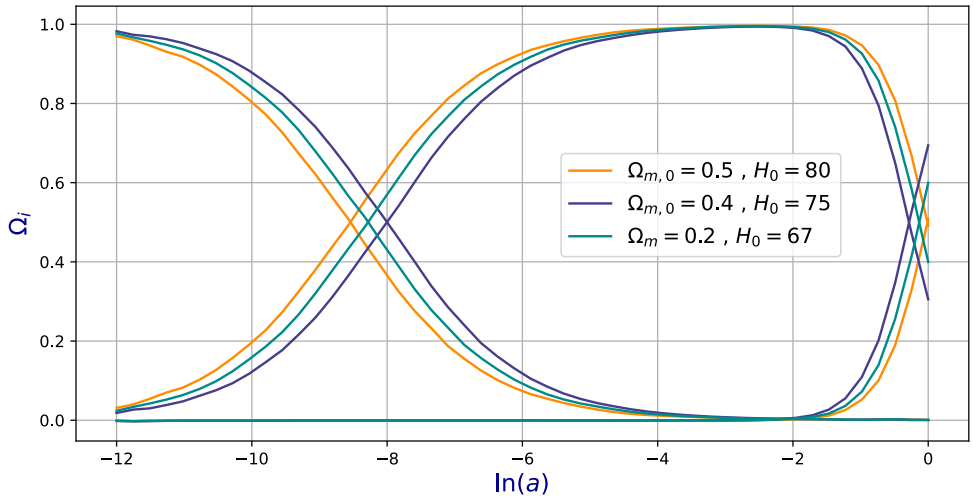


Figura 6.10: Soluciones generadas por medio de la red neuronal simplemente evaluándola en distintas condiciones iniciales.

6.3 Clasificación de objetos interestelares

La clasificación de los objetos celestes es una de las principales tareas que los astrónomos han realizado a lo largo de la historia. El aumento del flujo de datos que se recibe del cosmos es a la vez una oportunidad y un reto. Permite clasificar los objetos en función de muchas de sus características, pero la cantidad de información es tan grande que puede convertirse en una tarea desalentadora. Este problema de investigación muestra que el aprendizaje profundo es una opción fantástica para abordar problemas de clasificación, en este caso la clasificación de objetos estelares. Para realizar esta tarea se utilizará un MLP, pero será necesario darle una nueva estructura a los datos de entrenamiento, introducir una nueva función de activación y definir otra función de coste. Supóngase que se debe clasificar objetos entre un determinado número de clases. Entonces, es necesario conocer una serie de características o atributos de los objetos que determinan a qué clase pertenecen, para luego mostrárselos al modelo y a su vez este sea capaz de hacer predicciones.

6.3.1 Fundamentos de la clasificación en el aprendizaje profundo

La clasificación más sencilla es la binaria, donde existen sólo dos clases. Para esto, procesar los atributos con la propagación hacia delante requiere de aplicar una función de activación en la última capa que permita dividir los elementos entre las clases correspondientes. Una primer propuesta podría ser una función escalón como la de la Figura 6.11, asignando a cada clase uno de los posibles valores, por ejemplo 0 y 1. Sin embargo, esta propuesta tiene ciertos inconvenientes: el primero es que una función de este tipo no es diferenciable en el punto donde se realiza el salto. Nótese que ya se ha mencionado una función de activación con esta particularidad: La función ReLU (Ecuación 4.6). Este problema no es tan grave mientras la función de activación reciba valores distintos al de la singularidad. El inconveniente grave radica en que una función escalón tiene una derivada nula en todos los puntos, por lo que es imposible propagar el error de la función de coste hacia las capas ocultas.

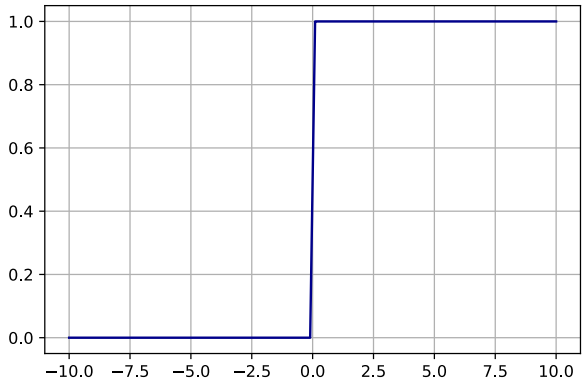


Figura 6.11: Función escalón.

Como solución a este problema es posible utilizar funciones de activación de un comportamiento parecido a un escalón, como sigmoide (4.8) o tangente hiperbólica (4.10). Estas funciones son diferenciables en todo su dominio y su derivada no se anula nunca. Por otro lado, aplicar la propagación hacia delante requiere que los atributos estén dados de forma numérica y las etiquetas (generalmente dadas por

variables categóricas) deben convertirse a una forma que puedan ser procesadas por la red. Aquí se considerará el caso más general, cuando se tiene un número arbitrario de clases.

$$(\text{atributo 1}, \text{atributo 2}, ..., \text{atributo k}) = \text{clase 1},$$

$$(\text{atributo 1}, \text{atributo 2}, ..., \text{atributo k}) = \text{clase 2},$$

$$(\text{atributo 1}, \text{atributo 2}, ..., \text{atributo k}) = \text{clase 3}.$$

Las etiquetas se transforman a lo que se denomina como vector *one-hot*. La transformación consiste en generar vectores de dimensión n, cuyas entradas ordenadas deberán corresponderse con las n clases con las que se cuente. En este caso $n = 3$, y el orden de las clases se puede definir sin pérdida de generalidad como:

$$(\text{clase 1},\ \text{clase 2},\ \text{clase 3})\,.$$

Así, a los elementos se les asigna un vector cuya entrada correspondiente a su clase será 1 y 0 en las demás:

$$(\text{atributo 1}, \text{atributo 2}, ..., \text{atributo k}) = (1, 0, 0),$$

$$(\text{atributo 1}, \text{atributo 2}, ..., \text{atributo k}) = (0, 1, 0),$$

$$(\text{atributo 1}, \text{atributo 2}, ..., \text{atributo k}) = (0, 0, 1).$$

En la estructura one-hot, es posible interpretar el valor de cada entrada como la probabilidad de que el objeto pertenezca a la clase correspondiente. Ahora, al procesar los atributos a través de la red ya es posible aplicarles una función de activación que asigne un valor a dicha probabilidad. La función más utilizada en procesos de clasificación es la conocida como *activación Softmax*, que aplicada en la última capa de la red, permite saber el grado de probabilidad de que el elemento que está procesando pertenezca a cada una de las clases. Aplicada a un vector x con k entradas, la función

softmax está definida como:

$$\text{softmax}(x) = \frac{e^{x_i}}{\sum_{j=1}^{k} e^{x_j}}. \tag{6.3}$$

Aplicada a la salida de la red, se obtiene un vector $a^L = \text{softmax}(z^L)$ que con entradas $a_i^L = \frac{e^{z_i^L}}{\sum_j e^{z_j^L}}$. Haciendo un poco de aritmética es fácil comprobar que:

- $0 \leq a_i^L \leq 1.$
- $\sum_j a_i^L = 1.$

Por lo que la función softmax es formalmente una distribución de probabilidad, así que efectivamente es posible interpretar las componentes de la salida como tal. Nótese que para el caso en que $n = 2$, softmax regresa a la forma de la función sigmoide que se mencionó en la clasificación binaria.

$$a_i^L = \frac{1}{1 + e^{(a_i^L - a_j^L)}},\ j \neq i. \tag{6.4}$$

La métrica para comparar las salidas de la red con las etiquetas es la función de coste llamada *Cross-Entropy*. Suponiendo que se tienen K clases distintas, esta función se define como:

$$C(Y, a^L) = -\sum_{j=1}^{K} Y_j \ln(a_j^L). \tag{6.5}$$

Donde se asume que las etiquetas Y se han transformado a vectores one-hot. Como Y_i vale 1 en la clase a la que pertenece (supóngase que es j) y 0 en las demás, la función (6.5) es ahora:

$$C(Y, a^L) = -\ln(a_j^L). \tag{6.6}$$

Ahora, para calcular la probabilidad total del conjunto de entrenamiento X, Y con n elementos, se tiene:

$$P\left(X|Y\right) = \prod_{i=1}^{n} P\left(X_i|Y_i\right). \tag{6.7}$$

Así, el objetivo será maximizar la Ecuación 6.7. Aplicando el logaritmo natural negativo de ambos lados de la igualdad:

$$-\ln(P(X|Y)) = \sum_{i=1}^{n} -\ln(P(x_i|\, y_i)). \tag{6.8}$$

La probabilidad P será la entregada por el modelo, por lo que $P\,(x_i|\, y_i) = a_j^L$. Así que, usando la definición de la Ecuación (6.6):

$$-\ln(P(X|Y)) = \sum_{i=1}^{n}\{-\ln(a_j^L)\}_i = \sum_{i=1}^{n}\{C(Y, a^L)\}_i. \tag{6.9}$$

Por lo tanto, para maximizar las probabilidades que arroja el modelo, se debe minimizar la Ecuación (6.9), que representa la suma de la función cross-entropy aplicada a cada elemento de los datos X, Y.

6.3.2 Los objetos astronómicos a clasificar

El conjunto de datos utilizado en este ejemplo es el DR14, publicado por el *Sloan Digital Sky Survey*. Consiste en una serie de observaciones realizadas hasta 2016 de distintos objetos estelares: Estrellas, QSO y Galaxias [52]. El conjunto de datos divide todas las observaciones en esas tres clases, y se les asigna una serie de 18 atributos que incluyen:

- *objid*: el identificador del objeto.
- *ra* y *dec*: las coordenadas celestes.
- *u, g, r, i, z*: son las mediciones correspondientes a las 5 bandas del telescopio, en el sistema de magnitudes astronómicas de Thuan-Gunn.
- *run, rereun, camcol, field*: son respectivamente; el número de ejecución, que identifica el análisis específico del objeto; la columna de la cámara, un número del 1 al 6 que identifica la línea de exploración dentro de la captura; el número de campo, que generalmente comienza en 11 y puede llegar a 800 para ejecu-

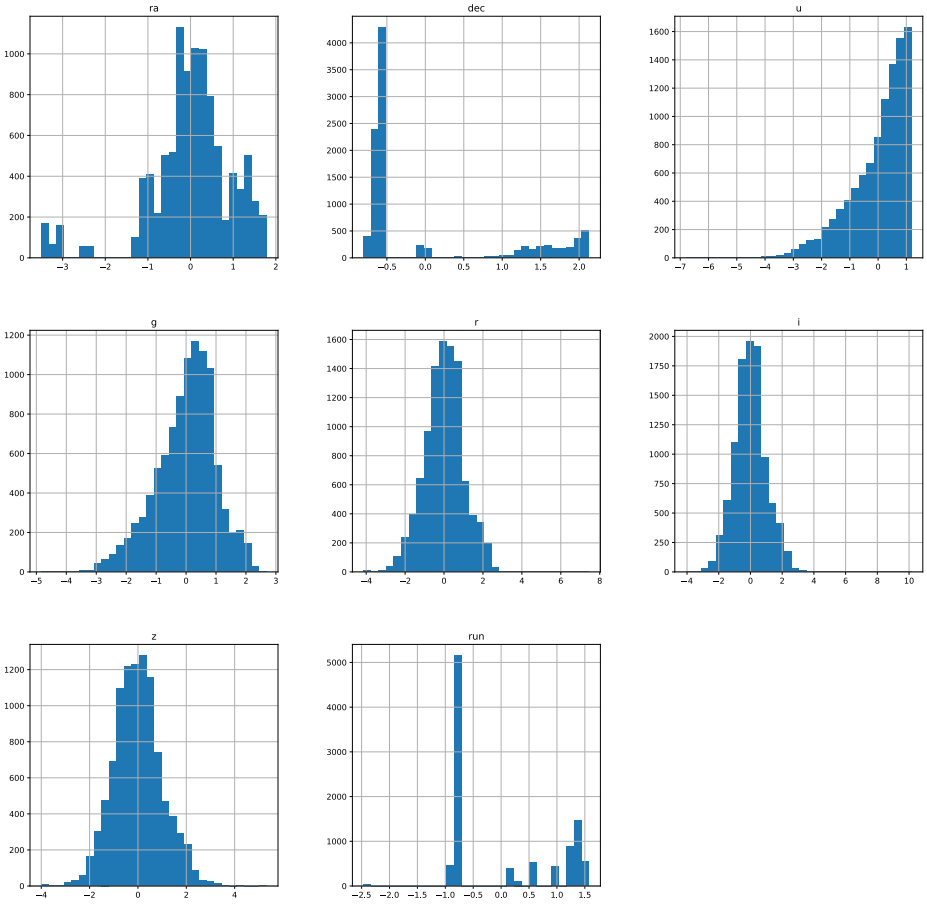


Figura 6.12: Histograma de los atributos utilizados para el entrenamiento del modelo de clasificación. A estas distribuciones ya se les ha aplicado una transformación de punto z.

ciones particularmente largas; y reejecutar, un parámetro que especifica cómo se procesó la imagen.

- *specobjid*: identificador de objeto.
- *clase*: indica la clase de objeto ya sea galaxia, estrella o cuásar.
- *redshift*: corrimiento al rojo.

- *plate*: número de placa.
- *mjd*: MJD de observación.
- *fiberid*: ID de fibra utilizada.

Varias de estas características son propias del instrumental del SDSS y no son observables del objeto estudiado, por lo que de entrada esas etiquetas sólo aportarían ruido al modelo de clasificación y deben ser removidas del conjunto de entrenamiento. Por otro lado, aunque el corrimiento al rojo sí es un observable del objeto, este parámetro aporta demasiada información al modelo y en la práctica uno espera hacer predicciones sobre las clases de los objetos antes de considerar obtener su z. Así que también debe ser descartado. Entonces, las etiquetas utilizadas para este modelo de clasificación son únicamente las bandas de color *u, g, r, i, z* y las coordenadas celestes *ra* y *de*.

En la Figura (6.12) se aprecia la distribución de las frecuencias de los atributos y sus rangos. A este conjunto ya se les aplicó una normalización de punto z para que todos estén dentro de un rango de valores equivalente y ninguno de ellos influya más que otro sobre el modelo a priori. Esos son todos los atributos que serán considerados por la red neuronal. Además de remover todos los atributos que no eran observables de los objetos, se realizó un análisis exhaustivo de correlación entre las variables con el fin de mostrar que la red puede procesar todos los atributos y encontrar un buen modelo predictivo.

Para esta tarea se designó una arquitectura de red compuesta por tres paredes ocultas, con 150 nodos y activación sigmoide en cada capa; finalmente, una capa de salida donde se le implementó la función de activación softmax (Figura 6.13). Esta arquitectura fue lo suficientemente robusta para tratar con las características del dataset, el cual se dividió en dos partes: 8,000 datos para el aprendizaje y 2,000 reservados como un conjunto de prueba para el modelo ya entrenado.

Como se mencionó, la métrica de error utilizada en esta ocasión fue cross-entropy. Al proceso de aprendizaje se le asignaron 80 épocas, el conjunto de entrenamiento se dividió en 80 % para aprendizaje y 20 % para validación. El modelo alcanzó un 95 %

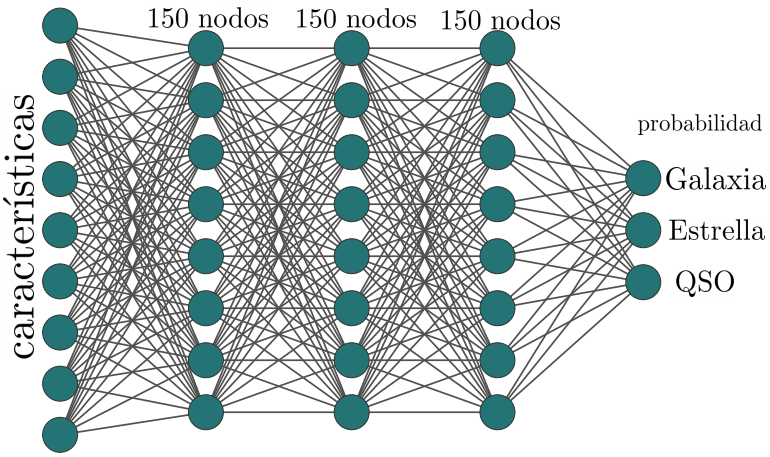


Figura 6.13: Arquitectura de ANN utilizada en la clasificación de objetos estelares, en las capas ocultas se aplicó la función de activación sigmoide, mientras que en la capa de salida se utilizó la función softmax.

de exactitud con sus datos de entrenamiento y un 94 % con los de validación (Figura 6.14). Donde la exactitud se calcula como el cociente de las clases acertadas entre el número total de predicciones.

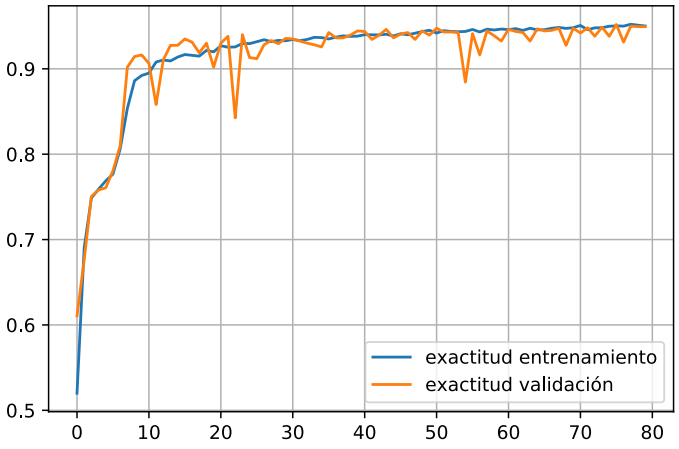


Figura 6.14: Exactitud durante las épocas.

Por último se probó la eficacia del modelo pidiéndole que clasificara el conjunto de prueba, pues como se explicó en la sección 3, al ser ajenos a su proceso de aprendizaje, este determinará si el modelo es capaz de generalizar lo aprendido. La red mostró ser

un éxito en este conjunto también, pues logró hacer la clasificación con un 94 % de exactitud.

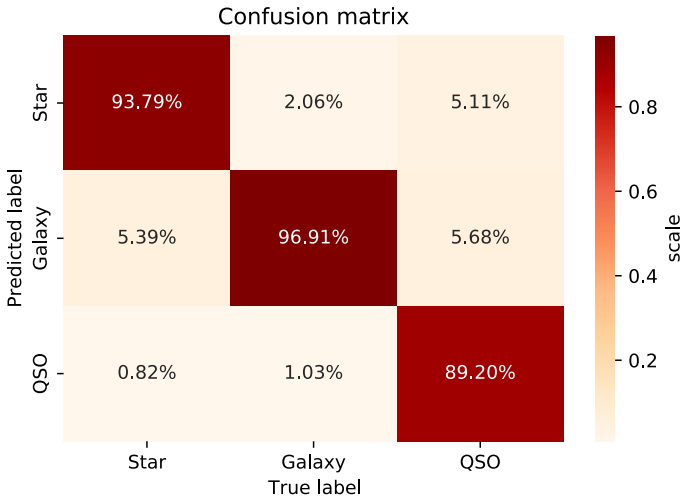


Figura 6.15: Matriz de confusión de las predicciones hechas por la red sobre el conjunto de prueba. Se muestran los porcentajes correspondientes que la red asignó a cada clase, donde el 100 % corresponde a la suma de los porcentajes por columna.

Esta información se organiza en forma de matriz de confusión (Figura 6.15), el cual es un arreglo que permite ver los errores o confusiones cometidos por el modelo y sus aciertos. Esto se hace relacionando los valores reales en el eje x y los predichos por el modelo en el eje y. Observe que los errores son mínimos en comparación a los aciertos, que se encuentran en la diagonal de la matriz. También se puede considerar a los elementos que conforman cada clase como un 100 %, así al hacer las predicciones se le pueden asignar distintos porcentajes según el número de aciertos o errores que cometió el modelo. Debido a que la red obtuvo muy buenos resultados en sus predicciones, se decidió incluir todos los atributos mostrados en la figura 6.12. Sin embargo, se realizaron distintas pruebas extrayendo características que pudieran estar separadas en clases a priori, es decir cuya distribución pareciera dividirse en tres grupos. Los modelos obtenidos en estas pruebas tuvieron resultados muy similares a los aquí presentados, por lo que se decidió no reportarlos con el fin de que esta sección fuese lo más concisa posible.

En conclusión, las redes neuronales son una buena opción para llevar a cabo este

tipo de tareas. Esto puede repercutir de forma positiva tanto en el tiempo como el esfuerzo implicado en clasificar objetos en astronomía, además de que permitiría observaciones más eficientes cuando se requiera estudiar un tipo de objeto en especial, pues un modelo entrenado puede hacer clasificaciones en tiempo real.

7 | Conclusiones

En este trabajo se presentaron algunos de los conceptos fundamentales de la cosmología, así como varios de los retos actuales que tiene que afrontar, desde el punto de vista computacional y de manejo de datos. También se propuso el uso de las redes neuronales artificiales como una herramienta capaz de solventar varios de esos retos. Con el fin de fundamentar tal propuesta, se desarrollaron tres problemas de investigación, donde se mostró cómo las técnicas del aprendizaje profundo pueden ser adoptadas por la cosmología. A través de dichas investigaciones se mostró que:

- Las redes neuronales tienen la capacidad de emular cualquier función o patrón que determine el comportamiento de un conjunto de datos determinado. Esto puede ser muy útil en diversas áreas científicas, ya que dichas redes pueden generar un modelo computacional para los datos y es una excelente alternativa cuando no se dispone de un modelo analítico satisfactorio.

- Una red neuronal debidamente entrenada puede utilizarse para sustituir los cálculos computacionales tradicionales en una amplia variedad de problemas y así disminuir el tiempo de cómputo requerido. Además, en la sección 6.2 se mostró que la red neuronal también proporciona un modelo que puede ser evaluado y manipulado matemáticamente, algo que los métodos numéricos tradicionales no siempre ofrecen.

- Con el último ejemplo, se demostró la gran eficacia de las redes neuronales en tareas que pueden ser complicadas de realizar por una persona en la práctica, -por ejemplo, la clasificación de objetos-. En este caso, se realizó una clasificación con atributos numéricos, obteniendo una excelente tasa de aciertos. Sin

embargo, la literatura indica que las redes neuronales también son una gran herramienta en la clasificación de imágenes y videos.

Por lo anterior, las redes neuronales mostraron ser una herramienta poderosa y versátil, cuya integración a la cosmología trae diversos beneficios. Así que la hipótesis de la que partió este trabajo ha sido comprobada de forma satisfactoria.

Por otro lado, aunque las RNA son consideradas como una gran herramienta para diversos problemas, en muchos casos pueden presentar algunas desventajas que el lector debe tener en cuenta. Por ejemplo: el tiempo de entrenamiento puede ser considerablemente largo para una base de datos muy grande o una arquitectura de red muy robusta. También se suele decir que las redes neuronales son 'cajas negras', en el sentido de que su enorme número de parámetros es sólo un conjunto de números reales sin información sobre los fenómenos que modelan, a diferencia de la información matemática que implicará una interpolación, por ejemplo. Otra consideración es que la RNA debe utilizarse con precaución para evitar el sobreajuste o el infraajuste y garantizar la generalización de lo aprendido del modelo y así evitar sesgos al hacer predicciones.

Finalmente, los tres problemas de investigación presentados en este trabajo son solo una pequeña muestra del potencial que el aprendizaje profundo puede desplegar en cosmología. Es una rama de la inteligencia artificial que está en auge y que, poco a poco, se está incorporando a diversas disciplinas científicas. Siguiendo con esta línea de trabajo, también se podrían presentar más herramientas de aprendizaje profundo para facilitar la gestión de datos en ciencia, principalmente en astronomía y cosmología observacional. Un siguiente peldaño podría encontrarse en las simulaciones numéricas del Universo y su contenido o el uso de redes neuronales convolucionales en tareas de clasificación o de simulaciones numéricas.

Bibliografía

[1] Juan de Dios Rojas Olvera, Isidro Gómez-Vargas, and Jose Alberto Vázquez. Observational cosmology with artificial neural networks. *Universe*, 8(2):120, 2022.

[2] Chester Bell, Tony Hey, and Alexander Szalay. Beyond the data deluge (computer science). *Science (New York, N.Y.)*, 323:1297–8, 04 2009.

[3] The Collaboration, Kazunori Akiyama, A. Alberdi, Walter Alef, Keiichi Asada, Rebecca Azulay, Anne-Kathrin Baczko, David Ball, Mislav Baloković, John Barrett, Dan Bintley, Lindy Blackburn, Wilfred Boland, Katherine Bouman, Geoffrey Bower, Michael Bremer, Christiaan Brinkerink, Roger Brissenden, Silke Britzen, and Paul Yamaguchi. First m87 event horizon telescope results. iii. data processing and calibration. *The Astrophysical Journal Letters*, 875, 04 2019.

[4] Charles Q. Choi. Migrating big astronomy data to the cloud. *Nature 584, 159-160*, 2020.

[5] Oliver James, Eugénie von Tunzelmann, Paul Franklin, and Kip S Thorne. Gravitational lensing by spinning black holes in astrophysics, and in the movie interstellar. *Classical and Quantum Gravity*, 32(6):065001, feb 2015.

[6] Mark Vogelsberger, Federico Marinacci, Paul Torrey, and Ewald Puchwein. Cosmological simulations of galaxy formation, 2019.

[7] Volker Springel, Rüdiger Pakmor, Annalisa Pillepich, Rainer Weinberger, Dylan Nelson, Lars Hernquist, Mark Vogelsberger, Shy Genel, Paul Torrey, Federico Marinacci, and Jill Naiman. First results from the IllustrisTNG simulations:

matter and galaxy clustering. *Monthly Notices of the Royal Astronomical Society*, 475(1):676–698, 12 2017.

[8] Hector J Hortua, Riccardo Volpi, Dimitri Marinelli, and Luigi Malago. Accelerating mcmc algorithms through bayesian deep networks. *arXiv preprint arXiv:2011.14276*, 2020.

[9] Isidro Gómez-Vargas, Ricardo Medel Esquivel, Ricardo García-Salcedo, and J Alberto Vázquez. Neural network within a bayesian inference framework. *J. Phys. Conf. Ser.*, 1723(1):012022, 2021.

[10] Alessio Spurio Mancini, Davide Piras, Justin Alsing, Benjamin Joachimi, and Michael P. Hobson. *CosmoPower* : emulating cosmological power spectra for accelerated bayesian inference from next-generation surveys, 2021.

[11] Celia Escamilla-Rivera, Maryi A Carvajal Quintero, and Salvatore Capozziello. A deep learning approach to cosmological dark energy models. , 2020(03):008, 2020.

[12] Guo-Jian Wang, Xiao-Jiao Ma, Si-Yao Li, and Jun-Qing Xia. Reconstructing functions and estimating parameters with artificial neural networks: A test with a hubble parameter and sne ia. , 246(1):13, 2020.

[13] Isidro Gómez-Vargas, J Alberto Vázquez, Ricardo Medel Esquivel, and Ricardo García-Salcedo. Cosmological reconstructions with artificial neural networks. *arXiv preprint arXiv:2104.00595*, 2021.

[14] C Baccigalupi, L Bedini, C Burigana, G De Zotti, A Farusi, D Maino, M Maris, F Perrotta, E Salerno, L Toffolatti, et al. Neural networks and the separation of cosmic microwave background and astrophysical signals in sky maps. *Monthly Notices of the Royal Astronomical Society*, 318(3):769–780, 2000.

[15] Matthew A. Petroff, Graeme E. Addison, Charles L. Bennett, and Janet L. Weiland. Full-sky cosmic microwave background foreground cleaning using machine learning. *The Astrophysical Journal*, 903(2):104, Nov 2020.

[16] J. Pasquet-Itam and J. Pasquet. Deep learning approach for classifying, detecting and predicting photometric redshifts of quasars in the sloan digital sky survey stripe 82. *Astronomy Astrophysics*, 611:A97, Mar 2018.

[17] I.E. Lagaris, A. Likas, and D.I. Fotiadis. Artificial neural networks for solving ordinary and partial differential equations. *IEEE Transactions on Neural Networks*, 9(5):987–1000, 1998.

[18] Satyasaran Changdar and Snehangshu Bhattacharjee. Solution of definite integrals using functional link artificial neural networks, 2019.

[19] Barbara Ryden. *Introduction to cosmology*. 2003.

[20] Silvio Bonometto, Vittorio Gorini, and Ugo Moschella. Modern cosmology. *Modern Cosmology. Series: Series in High Energy Physics, Cosmology and Gravitation, ISBN: 978-0-7503-0810-6. Taylor Francis, Edited by Ugo Moschella, Vittorio Gorini and Silvio Bonometto*, 01 2002.

[21] Bernard Jones. A brief history of cosmology. 126:1–10, 08 1997.

[22] Andrew R. Liddle. *An introduction to modern cosmology*. 1998.

[23] Adam G. Riess, Alexei V. Filippenko, Peter Challis, Alejandro Clocchiatti, Alan Diercks, Peter M. Garnavich, Ron L. Gilliland, Craig J. Hogan, Saurabh Jha, Robert P. Kirshner, and et al. Observational evidence from supernovae for an accelerating universe and a cosmological constant. *The Astronomical Journal*, 116(3):1009–1038, Sep 1998.

[24] Ofer Lahav and Andrew R Liddle. The cosmological parameters (2019), 2019.

[25] Daniel Aloni, Asher Berlin, Melissa Joseph, Martin Schmaltz, and Neal Weiner. A Step in Understanding the Hubble Tension. 10 2021.

[26] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[27] Frank Rosenblatt and S Papert. The perceptron. *A perceiving and recognizing automation, Cornell Aeronautical Laboratory Report*, pages 85–460, 1957.

[28] Marvin Minsky and Seymour Papert. Perceptron: an introduction to computational geometry. *The MIT Press, Cambridge, expanded edition*, 19(88):2, 1969.

[29] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[30] Héctor Calvo Pardo, Jose Olmo, and Tullio Mancini. Neural network models for empirical finance. *Journal of Risk and Financial Management*, 13, 10 2020.

[31] Paul Covington, Jay Adams, and Emre Sargin. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*, New York, NY, USA, 2016.

[32] Oran Gafni, Lior Wolf, and Yaniv Taigman. Live face de-identification in video, 2019.

[33] Travers Ching, Daniel S. Himmelstein, Brett K. Beaulieu-Jones, Alexandr A. Kalinin, Brian T. Do, Gregory P. Way, Enrico Ferrero, Paul-Michael Agapow, Michael Zietz, Michael M. Hoffman, Wei Xie, Gail L. Rosen, Benjamin J. Lengerich, Johnny Israeli, Jack Lanchantin, Stephen Woloszynek, Anne E. Carpenter, Avanti Shrikumar, Jinbo Xu, Evan M. Cofer, Christopher A. Lavender, Srinivas C. Turaga, Amr M. Alexandari, Zhiyong Lu, David J. Harris, Dave DeCaprio, Yanjun Qi, Anshul Kundaje, Yifan Peng, Laura K. Wiley, Marwin H. S. Segler, Simina M. Boca, S. Joshua Swamidass, Austin Huang, Anthony Gitter, and Casey S. Greene. Opportunities and obstacles for deep learning in biology and medicine. *Journal of The Royal Society Interface*, 15(141):20170387, 2018.

[34] Niels Justesen, Philip Bontrager, Julian Togelius, and Sebastian Risi. Deep learning for video game playing, 2019.

[35] Diego Andina, D. Pham, D. Andina, Antonio Vega-Corona, Juan Seijas, and J. Torres-Garcìa. *Neural Networks Historical Review*. 02 2007.

[36] Robert M Gower. Convergence theorems for gradient descent. *Lecture notes for Statistical Optimization*, 2018.

[37] Edwin Hubble. A relation between distance and radial velocity among extra-galactic nebulae. *Proceedings of the National Academy of Sciences*, 15(3):168–173, 1929.

[38] Jasjeet Bagla. Hubble, hubble's law and the expanding universe. *Resonance*, 14:216–225, 03 2009.

[39] Wendy L. Freedman, Barry F. Madore, Brad K. Gibson, Laura Ferrarese, Daniel D. Kelson, Shoko Sakai, Jeremy R. Mould, Jr. Robert C. Kennicutt, Holland C. Ford, John A. Graham, John P. Huchra, Shaun M. G. Hughes, Garth D. Illingworth, Lucas M. Macri, and Peter B. Stetson. Final results from theHubble space TelescopeKey project to measure the hubble constant. 553(1):47–72, may 2001.

[40] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[41] Xue Ying. An overview of overfitting and its solutions. *Journal of Physics: Conference Series*, 1168:022022, feb 2019.

[42] Haider Allamy. Methods to avoid over-fitting and under-fitting in supervised machine learning (comparative study). 12 2014.

[43] Aston Zhang, Zachary C. Lipton, Mu Li, and Alexander J. Smola. Dive into deep learning, 2021.

[44] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[45] Christos Louizos, Max Welling, and Diederik P Kingma. Learning sparse neural networks through l_0 regularization. *arXiv preprint arXiv:1712.01312*, 2017.

[46] Ekachai Phaisangittisagul. An analysis of the regularization between l2 and dropout in single hidden layer neural network. In *2016 7th International Conference on Intelligent Systems, Modelling and Simulation (ISMS)*, pages 174–179. IEEE, 2016.

[47] Sotiris Kotsiantis, Dimitris Kanellopoulos, and P. Pintelas. Data preprocessing for supervised learning. *International Journal of Computer Science*, 1:111–117, 01 2006.

[48] Salvador García, Sergio Ramírez-Gallego, Julián Luengo, José Benítez, and Francisco Herrera. Big data preprocessing: methods and prospects. *Big Data Analytics*, 1, 11 2016.

[49] Luis A. Escamilla and J. Alberto Vazquez. Model selection applied to nonparametric reconstructions of the Dark Energy. 11 2021.

[50] Vishnu Kolisetty and Dharmendra Rajput. A review on the significance of machine learning for data analysis in big data. *Jordanian Journal of Computers and Information Technology*, 06:1, 01 2019.

[51] Código completo. https://github.com/JuanDDiosRojas/Arts/tree/main/Deep%20Learning%20and%20its%20applications%20to%20cosmology. Publicado: enero/2022.

[52] Dr14. https://www.sdss.org/dr14/.