

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE INFORMÁTICA
SANTIAGO - CHILE



“DETECCIÓN DE POSIBLES EXOPLANETAS MEDIANTE DEEP LEARNING EN DATOS DE MISIONES ESPACIALES CON NANOSATÉLITES”

ÁLVARO FÉLIX FUENTES MONDACA

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL EN INFORMÁTICA

Profesor Guía: Mauricio Solar
Profesor Correferente: Ricardo Ñanculef

Julio - 2023

DEDICATORIA

A mi familia que me apoya, a mis amigos que me acompañaron y a todas las personas inmersas en la búsqueda de nuevos mundos.

RESUMEN

Resumen— En este trabajo se estudia la detección de posibles exoplanetas utilizando modelos de *Deep Learning* sobre los datos de la misión BRITE, una constelación de 5 nanosatélites. Se crea un *dataset* de curvas de luz con señales sintéticas mediante la aplicación de un pre-procesamiento a los datos de BRITE y una inyección de tránsitos según el modelo Mandel y Agol con una restricción de radio estelar $< 3.08 [R_{sun}]$ y radio planetario entre 0.95 y 2.1 $[R_{jup}]$. Aplicando un criterio de calidad se obtienen 597 ejemplos *Planet Candidate* (PC) y 3126 ejemplos no candidato, los cuales se utilizan para entrenar un modelo modificado de Red Neuronal Convolutiva AstroNet, obteniendo sobre un conjunto de *testing* las métricas 100 % de precisión y 96.72 % recall para la clase PC, y AUC-ROC de 98.36 %. Finalmente se buscan posibles exoplanetas en los datos pre-procesados de BRITE, encontrando señales similares a tránsitos en los objetivos HD 039060, HD 218396, HD 022049 y HD 036861. Estas señales debe ser investigadas en un trabajo futuro.

Palabras Clave— exoplanetas, deep learning, nanosatélites, BRITE.

ABSTRACT

Abstract— In this work, the detection of possible exoplanets using *Deep Learning* models is studied on data from the BRITE mission, a constellation of 5 nanosatellites. A *dataset* of light curves with synthetic signals is created by applying a pre-processing to the BRITE data and an injection of transits according to the Mandel and Agol model with a constraint of stellar radius $< 3.08 [R_{sun}]$ and planetary radius between 0.95 and 2.1 $[R_{jup}]$. Applying a quality criteria, 597 *Planet Candidate*(PC) examples and 3126 non candidate examples are obtained, which are used to train a modified AstroNet Convolutional Neural Network model, obtaining over a *testing* set the metrics 100 % precision and 96.72 % recall for the PC class, and AUC-ROC of 98.36 %. Finally, possible exoplanets are searched in the pre-processed data from BRITE, finding signals similar to exoplanetary transits in the targets HD 039060, HD 218396, HD 022049 and HD 036861. These signals should be investigated in future work.

Keywords— exoplanets, deep learning, nanosatellites, BRITE.

GLOSARIO

ANN: Artificial Neural Network.
ASTERIA: Arcsecond Space Telescope Enabling Research in Astrophysics.
AUC-ROC: Area Under the Curve Receiver Operating Characteristics.
BAb: BRITE-Austria b.
BHr: BRITE-Heweliusz r.
BMb: BRITE-Montreal b.
BRITE: BRiGht-star Target Explorer.
BTr: BRITE-Toronto r.
CADARS: Catalogue of Stellar Diameters.
CCD: Charge Coupled Device.
CNN: Convolutional Neural Network.
CTL: Candidate Target List.
CUTE: Colorado Ultraviolet Transit Experiment.
DR: Data Release.
FoV: Field of View.
GESD: Generalized Extreme Studentized Deviate.
GRB: Gamma-ray burst.
HJD: Heliocentric Julian Date.
KIC: Kepler Input Catalog.
LSP: Lomb–Scargle Periodogram.
LSTM: Long short-term memory.
MAST: Mikulski Archive for Space Telescopes.
MLP: Multi-layer Perceptron.
MP: Megapixel.
NASA: National Aeronautics and Space Administration.
PC: Planet Candidate.
PDCSAP: Pre-search Data Conditioning SAP.
PSF: Point-Spread Function.
SPARCS: Star-Planet Activity Research CubeSat.
TCE: Threshold Crossing Event.
TESS: Transiting Exoplanet Survey Satellite.
TIC: Tess Input Catalog.
UBr: UniBRITE r.

ÍNDICE DE CONTENIDOS

RESUMEN	III
ABSTRACT	III
GLOSARIO	IV
ÍNDICE DE FIGURAS	VII
ÍNDICE DE TABLAS	X
INTRODUCCIÓN	1
CAPÍTULO 1: DEFINICIÓN DEL PROBLEMA	3
1.1 LA BÚSQUEDA DE EXOPLANETAS	3
1.2 EL COSTO DE LAS MISIONES DE GRAN ENVERGADURA Y EL USO DE NANOSATÉLITES	3
1.3 OBJETIVOS	6
1.3.1 OBJETIVO GENERAL	6
1.3.2 OBJETIVOS ESPECÍFICOS	6
CAPÍTULO 2: MARCO CONCEPTUAL	7
2.1 DETECCIÓN DE EXOPLANETAS	7
2.1.1 MÉTODO DEL TRÁNSITO	7
2.1.2 ANÁLISIS DE CURVAS DE LUZ	8
2.1.3 REPRESENTACIÓN DE CURVAS DE LUZ DE TRÁNSITOS	12
2.2 MISIONES DE BÚSQUEDA DE EXOPLANETAS	17
2.2.1 MISIONES TRADICIONALES	17
2.2.2 ASTRONOMÍA CON NANOSATÉLITES	20
2.2.3 MISIÓN BRITTE	21
2.2.4 MISIÓN ASTERIA	25
2.3 MODELOS DE DEEP LEARNING	26
2.3.1 INTRODUCCIÓN	26
2.3.2 ARQUITECTURAS	28
2.3.3 ENTRENAMIENTO	31
2.3.4 MÉTRICAS, REGULARIZACIÓN Y TASAS ADAPTATIVAS	35
2.3.5 APLICACIONES	37
CAPÍTULO 3: PROPUESTA DE SOLUCIÓN	42
3.1 PREPARACIÓN	42
3.1.1 LIBRERÍAS DISPONIBLES	42
3.1.2 DATOS DISPONIBLES	43
3.1.3 PREPROCESAMIENTO	44

3.2	CREAR DATASET	53
3.2.1	PREPARACIÓN	53
3.2.2	INYECCIÓN	56
3.3	APLICACIÓN DE MODELOS	61
3.3.1	SELECCIÓN Y ENTRENAMIENTO	61
3.3.2	BÚSQUEDA DE POSIBLES EXOPLANETAS	64
CAPÍTULO 4: VALIDACIÓN DE LA SOLUCIÓN		69
4.1	VALIDAR EL PREPROCESAMIENTO	69
4.2	VALIDAR EL DATASET	71
4.3	VALIDACIÓN DE MODELOS	72
CAPÍTULO 5: CONCLUSIONES		75
5.1	TRABAJO FUTURO	78
ANEXOS		79
6.1	ALGORITMOS DE TASAS ADAPTATIVAS	79
6.2	CASOS ESPECIALES EN BRITE	81
6.3	RADIOS ESTELARES	82
REFERENCIAS BIBLIOGRÁFICAS		84

ÍNDICE DE FIGURAS

1.1	Ilustración de una detección de tránsito. Fuente: NASA.	3
1.2	Comparación de tamaños de un CubeSat 2U (LUME 1) y el satélite de comunicaciones Hispasat 36W-1 (paneles solares removidos por elaboración propia). Fuente: [Alen, 2021]	4
1.3	Árbol del problema de causa-problema principal-efecto. Fuente: elaboración propia	5
2.1	(izquierda) una curva de luz del sistema WASP-126 por la misión TESS. Los puntos de PDCSAP Flux que descienden periódicamente corresponden al tránsito del exoplaneta WASP 126-b. (derecha) acercamiento del tránsito centrado en el tiempo 1327.520678 [days]. Fuente: Scott W. Fleming, STScI: https://spacetelescope.github.io/notebooks/notebooks/MAST/TESS/beginner_how_to_use_lc/beginner_how_to_use_lc.html	8
2.2	Curva de luz de una estrella variable durante 600 noches consecutivas. Fuente: [Cryer y Chan, 2008]	10
2.3	Periodogramas calculados sobre la curva de luz de la estrella variable. Fuente: [Cryer y Chan, 2008]	10
2.4	Ejemplo de <i>detrending</i> . (superior) Curva original. (inferior) <i>Detrending</i> aplicado sobre la curva con método <i>biweight</i> . Fuente: [Hippke <i>et al.</i> , 2019]	11
2.5	(izquierda) curva <i>phase-folded</i> del sistema WASP-126 utilizando el periodo de tránsito del exoplaneta WASP-126 b, usando los mismos datos de la Figura 2.1. (derecha) curva <i>phase-folded</i> usando un periodo distinto del tránsito. El valor de epoch es el mismo para ambas curvas (1327.520678 [days]). Fuente: Elaboración propia.	13
2.6	Vista global y local de la curva <i>phase-folded</i> centrada en un TCE del objetivo TESS Input Catalog (TIC) 25155310. En esta curva hay un exoplaneta confirmado. Fuente: elaboración propia utilizando los datos ya procesados de [Yu <i>et al.</i> , 2019].	14
2.7	Efecto de <i>limb darkening</i> en nuestro sol. Fuente: http://spiff.rit.edu/classes/phys440/lectures/limb/limb.html	15
2.8	Simulación de curva de luz usando el algoritmo de Mandel y Agol. A la señal se le aplica ruido gaussiano con media cero y desviación estándar 1. Fuente: elaboración propia.	16

2.9	Descubrimientos de exoplanetas confirmados por misión/instalación usando el método del tránsito (23/07/2022)). Fuente: elaboración propia usando datos de NASA Exoplanet Archive.	17
2.10	Esquema de un nanosatélite BRITE, utilizado para fotometría de estrellas brillantes. Fuente: [Weiss et al., 2014]	22
2.11	Efecto de la técnica “chopping”. Los subraster izquierdo y medio corresponden a exposiciones consecutivas de la estrella HD23630, ambos con un PSF estelar dominante. A la derecha se aprecia la resta de ambos subrasters, donde los defectos de background son removidos. Fuente: [Pablo et al., 2016]	24
2.12	Ejemplo de neurona. Fuente: Elaboración propia.	27
2.13	Red neuronal con arquitectura FeedForward densa. Las capas ocultas van desde $a^{(1)}$ a $a^{(L-1)}$ Fuente: Elaboración propia.	28
2.14	(superior) Ejemplo de capa convolucional 1D con $K = 3, S = 1$. (inferior) Capa convolucional 1D con $K = 3, S = 2$. En ambos el input es de largo 7. Fuente: Elaboración propia.	30
2.15	Ejemplo de capa convolucional ($K = 3$) con pesos compartidos y 3 filtros. El número de parámetros entrenables es 12 considerando los 3 bias asociados. Fuente: Elaboración propia.	31
2.16	Arquitectura implementada para la tarea de detección de tránsitos con periodo de 1 a 2 días. Fuente: [Yeh y Jiang, 2020]	39
2.17	Arquitectura implementada en la tarea de triaje. Fuente: [Yu et al., 2019]	41
3.1	Dos curva de luz resultantes del reformato de los <i>raw fluxes</i> y el tiempo. Las estrellas corresponden a HD92287 y HD218396. Fuente: Elaboración propia.	46
3.2	Ejemplos de outliers (marcados con círculos rojos) para las columnas 'Magnitude', 'PSFC1', 'RTSC' y 'APERO'. Se utiliza el <i>sigma clipping</i> con valor de $\alpha = 3$. Fuente: Elaboración propia.	47
3.3	Ejemplos de <i>clusters</i> en la columna 'XCEN' e 'YCEN'. Los outliers son marcados con círculos rojos. Fuente: Elaboración propia.	47
3.4	Diagrama orbital del método <i>naive</i> propuesto. Fuente: Elaboración propia.	48
3.5	Ejemplo de distribución de datos por órbita en el método <i>naive</i> propuesto. Fuente: Elaboración propia.	49

3.6	Primeras cuatro iteraciones de la implementación usada en la decorrelación. En las iteraciones 1 y 3 no se realiza la corrección. El resultado final tras 20 iteraciones se presenta en la esquina inferior derecha. Fuente: Elaboración propia.	52
3.7	Ejemplos de tránsitos generados y sus representaciones Fold, Local y Global View. Los puntos color naranja corresponden al tránsito inyectado. Fuente: Elaboración propia.	60
3.8	Histogramas con los valores de los parámetros en los subconjuntos de entrenamiento y validación. Fuente: Elaboración propia.	62
3.9	<i>Loss</i> por <i>epoch</i> del modelo final (astronet_46). Fuente: Elaboración propia. .	63
3.10	Matriz de confusión y métricas del modelo final (astronet_46) sobre el Testing Set. Fuente: Elaboración propia.	64
3.11	Casos encontrados con señales similares a un exoplaneta en los datos de BRITE, junto a los parámetros predichos. Fuente: Elaboración propia.	66
3.12	Casos encontrados cuyas señales son similares a un exoplaneta detectadas en HD 218396 en <i>time epochs</i> cercanos. Fuente: Elaboración propia.	67
3.13	Casos encontrados cuya señal se asemeja a estrellas variables. Fuente: Elaboración propia.	68
4.1	(superior) Espectro de amplitud de Fourier de HR 8799 calculado por [Sódor y Bognár, 2020]. (inferior) Periodograma Lomb-Scargle generado con los datos pre-procesados. Fuente: [Sódor y Bognár, 2020] y elaboración propia.	70
4.2	Métricas del entrenamiento y evaluación del modelo AstroNet-Triage implementado en Keras. Fuente: Elaboración propia.	72
4.3	Matriz de confusión y métricas del modelo basado en AstroNet sobre conjunto de validación. Fuente: Elaboración propia.	73
4.4	Matriz de confusión y métricas del modelo basado en [Yeh y Jiang, 2020] sobre conjunto de validación. Fuente: Elaboración propia.	74

ÍNDICE DE TABLAS

2.1	Parámetros de entrada del algoritmo Mandel y Agol para producir la curva de la Figura 2.8. Fuente: elaboración propia.	16
2.2	características de los lentes de TESS. Fuente: [Ricker <i>et al.</i> , 2016]	19
2.3	Tabla de misiones con nanosatélites de interés dentro del trabajo de investigación. Fuente: elaboración propia.	21
2.4	Resumen de los 6 nanosatélites del proyecto BRITE. BMb no entró en operaciones debido a una falla en la separación del vehículo de lanzamiento. Fuente: elaboración propia. Datos orbitales de n2yo.com (20/05/2022)	23
2.5	Rango de valores usados para el algoritmo de Mandel y Agol. Fuente: [Yeh y Jiang, 2020]	38
3.1	Tabla con los valores de headers más relevantes para esta investigación. Fuente: elaboración propia.	44
3.2	Tabla con las columnas presentes en los datos. Los datos en DR6 poseen más columnas, pero no se utilizan durante el procesamiento de datos en este trabajo. Fuente: elaboración propia.	45
3.3	Rango de valores usados en la inyección de tránsito. Fuente: Elaboración propia.	57
6.1	Estrellas cuyo radio es menor o igual a 3.08 radio solar (Tabla 1/2). Fuente: Elaboración propia según lo descrito en el SubCapítulo 3.2.1.	82
6.2	Estrellas cuyo radio es menor o igual a 3.08 radio solar (Tabla 2/2). Fuente: Elaboración propia según lo descrito en el SubCapítulo 3.2.1.	83

INTRODUCCIÓN

La astronomía antes de la carrera espacial estaba limitada al estudio de la radiación electromagnética que es capaz de cruzar la atmósfera que nos separa del espacio exterior. Con los avances de la tecnología satelital se abrieron las puertas al estudio astronómico de frecuencias más altas, observaciones que no son posibles por la rotación de la Tierra y el estudio de luz menos distorsionada al no cruzar nuestra densa atmósfera. De esta forma, múltiples misiones espaciales astronómicas observan diversos fenómenos físicos a través de distintas longitudes de onda para complementar las observaciones terrestres.

Entre los fenómenos estudiados desde el espacio que han llevado a miles de descubrimientos se encuentra la detección de **exoplanetas**: planetas que orbitan estrellas diferentes al Sol. Un verdadero logro científico e ingenieril ya que son difíciles de detectar debido a sus tenues brillos respecto a las brillantes estrellas que orbitan. En este contexto, por lo general, se utilizan técnicas de detección indirecta para descubrir exoplanetas.

Entre estas técnicas se encuentra el **método del tránsito**, el cual consiste en estudiar variaciones de luz de una estrella en el tiempo para encontrar una señal característica que indique un planeta pasando entre la estrella y un telescopio observador, evento que es periódico. Un aspecto relevante es que se desconoce a-priori el periodo del evento, el instante temporal en que sucede y las características del planeta, lo que dificulta su detección por criterio experto sobre una gran cantidad de datos.

Otro aspecto relevante son los altos costos de diseño, desarrollo, lanzamiento y operación de los telescopios espaciales. En este contexto, durante los últimos años se ha explorado el uso de **nanosatélites** para misiones astronómicas debido a su rápido desarrollo y bajos costos respecto a grandes misiones.

Considerando estos aspectos, el tema de esta memoria es la tarea de detección de posibles exoplanetas en datos de misiones astronómicas que utilizan tecnología nanosatelital, en particular con técnicas y modelos de **Deep Learning**, las cuales permiten una búsqueda entre gran cantidad de datos.

El contenido del documento se estructura en un capítulo de **Marco Conceptual** que describe el método del tránsito, las técnicas de análisis de curvas de luz y su representación para ser la entrada de un modelo de detección. Luego se dedica un subcapítulo a la comparación de distintas misiones espaciales relacionadas a tránsitos exoplanetarios junto a una revisión de las misiones astronómicas con nanosatélites destacando la constelación de nanosatélites BRITE: la principal fuente de datos de este trabajo. El Marco Conceptual finaliza detallando los conceptos, terminología y aplicaciones de Deep Learning en el contexto, destacando el uso de Redes Neuronales Convolucionales (CNN) por [Yeh y Jiang, 2020] y [Yu et al., 2019] (modelo AstroNet).

Posteriormente se presenta un capítulo de **Propuesta de Solución**, en el cual se listan las librerías de interés en Python, los datos disponibles de la misión BRITE (curvas de luz), el pre-procesamiento que debe aplicarse y la creación de un *dataset* de curvas de luz con señales sintéticas de exoplanetas, inyectados sobre de los datos de BRITE.

La última sección de la Propuesta de Solución es la aplicación de modelos de Deep Learning sobre el *dataset* creado, lo que incluye la selección de modelos y entrenamiento del modelo final, finalizando con la búsqueda de posibles exoplanetas en los datos pre-procesados de BRITE.

En el siguiente capítulo se presenta la **Validación de Solución**, donde se valida el pre-procesamiento y el *dataset* creado bajo un criterio de calidad utilizando el modelo AstroNet y finalmente se validan los modelos implementados a través de la obtención de valores altos en métricas. El documento finaliza con un capítulo de **Conclusión** donde se recopilan los procedimientos y resultados más relevantes, junto a una lista de trabajo futuro.

CAPÍTULO 1

DEFINICIÓN DEL PROBLEMA

1.1. LA BÚSQUEDA DE EXOPLANETAS

Entre las investigaciones que más han proliferado durante los últimos años gracias a telescopios espaciales están los descubrimientos de miles de exoplanetas, los cuales son planetas que orbitan una estrella diferente al Sol, y que por lo tanto no pertenecen al sistema solar. Al igual que nuestro planeta, algunos exoplanetas descubiertos no se encuentran solos, sino que existen otros exoplanetas orbitando una misma estrella. Hasta la fecha (17/07/2023) NASA ha informado 5470 exoplanetas confirmados con datos de observatorios espaciales y terrestres [NASA, 2023]. Misiones espaciales como Kepler y su extensión K2 durante sus funcionamientos utilizaron la popular técnica del tránsito para la detección de exoplanetas. Esta técnica consiste en detectar cómo varía la luz llega de una estrella debido a que un exoplaneta pasa entre ella y el observatorio. El estudio del brillo de la estrella en el tiempo (curvas de luz), junto a datos como el tamaño y la distancia a su estrella permiten caracterizar dicho exoplaneta. En la Figura 1.1 se presenta una ilustración del funcionamiento de este método utilizando una curva de luz.

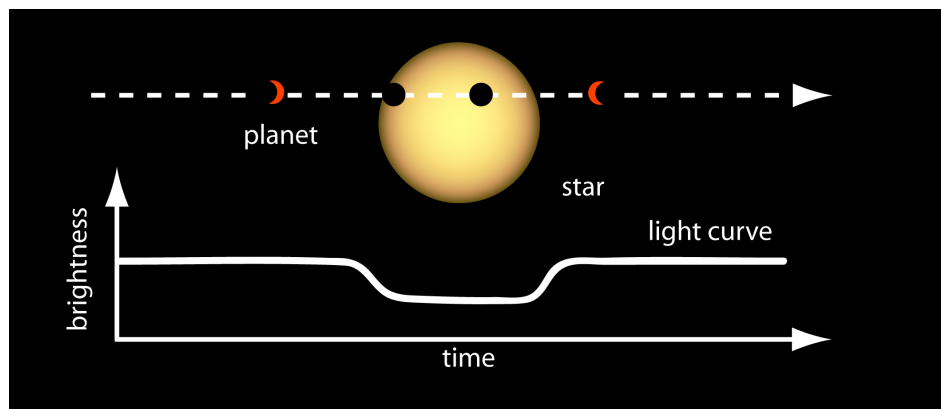


Figura 1.1: Ilustración de una detección de tránsito. Fuente: NASA.

1.2. EL COSTO DE LAS MISIONES DE GRAN ENVERGADURA Y EL USO DE NANOSATÉLITES

A pesar de los beneficios científicos del estudio con misiones espaciales, una gran desventaja que poseen respecto a observatorios terrestres son sus altos costos de diseño, desarrollo, lanzamiento y mantenimiento. Por ejemplo, *“el costo del ciclo de vida de la misión Kepler, incluyendo 3.5 años de operación y excluyendo el lanzamiento fue de aproximadamente \$600*

millones (de dólares)” [Serjeant *et al.*, 2020]. En este contexto, durante los últimos años se ha explorado el uso de nanosatélites para investigación científica en las áreas de astronomía y astrofísica. Por ejemplo, el estándar CubeSat es un tipo de nanosatélite que ha ganado popularidad en agencias espaciales y centros de investigación debido a su rápido desarrollo y bajos costos respecto a misiones de gran envergadura. El tamaño estándar de estos nanosatélites está construido en unidades de $10 \times 10 \times 10 [cm^3]$ llamado 1U, manteniendo un peso normalmente menor a 2 [Kg]. Figura 1.2 se presenta una comparación de tamaños del CubeSat 2U LUME 1 usado para monitorización de incendios forestales y el satélite de comunicaciones Hispasat 36W-1, el cual posee un tamaño similar a Kepler.

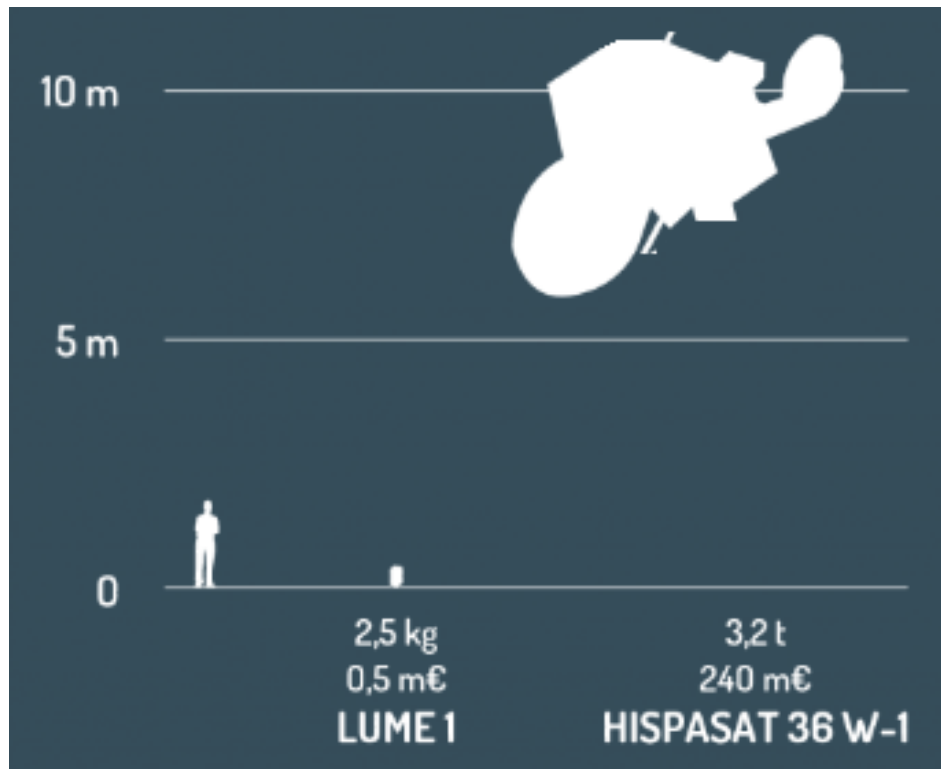


Figura 1.2: Comparación de tamaños de un CubeSat 2U (LUME 1) y el satélite de comunicaciones Hispasat 36W-1 (paneles solares removidos por elaboración propia). Fuente: [Alen, 2021]

La estandarización de CubeSat ha permitido la venta comercial de componentes estructurales, electrónicos, sistemas de energía y módulos de comunicación a un costo más accesible. A diferencia de los observatorios de gran envergadura que requieren una distribución del tiempo de observación entre muchos proyectos, un nanosatélite puede monitorear fuentes durante semanas o meses [Shkolnik, 2018], lo cual es beneficioso para investigaciones que busquen estudiar fenómenos particulares de gran interés.

Es por esto por lo que se busca estudiar el problema de detección de exoplanetas utilizando tecnología nanosatelital, en particular analizar la tarea de detección de posibles exoplanetas utilizando modelos y técnicas de *Deep Learning*. Las causas y efectos que motivan el uso

modelos de aprendizaje automático y nanosatélites se encuentran en la Figura 1.3. Entre las motivaciones de esta investigación se encuentran:

- Una gran cantidad de datos provenientes de observatorios futuros, tanto tradicionales como nanosatélites.
- La diversidad de fenómenos astronómicos que producen señales confundibles con tránsitos exoplanetarios como estrellas variables cefeidas, sistemas estelares binarios y estrellas pulsantes.
- Desconocimiento a priori del periodo de tránsito de los exoplanetas, lo que dificulta su detección por criterio experto sobre una gran cantidad de datos.
- Apoyar el trabajo sobre datos de las misiones astronómicas en curso y futuras que busquen exoplanetas con nanosatélites.

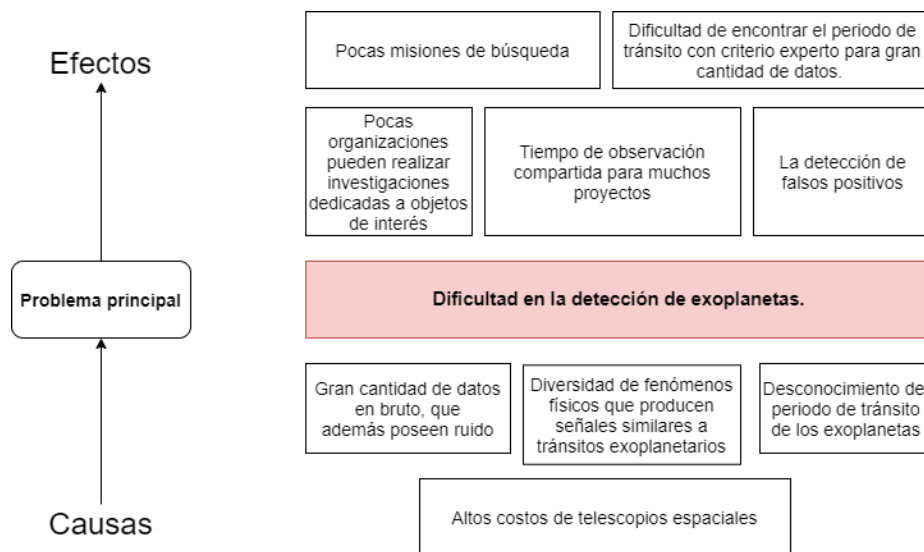


Figura 1.3: Árbol del problema de causa-problema principal-efecto. Fuente: elaboración propia

1.3. OBJETIVOS

1.3.1. OBJETIVO GENERAL

Analizar técnicas y modelos de *Deep Learning* en la tarea de detección de posibles exoplanetas en datos de misiones astronómicas que utilizan tecnología nanosatelital.

1.3.2. OBJETIVOS ESPECÍFICOS

- Comparar conjuntos de datos de curvas de luz provenientes de catálogos y surveys relacionados a tránsitos exoplanetarios.
- Investigar técnicas de representación de curvas de luz de tránsitos.
- Elegir modelos para representar curvas de luz que sirvan de entrada para la tarea de detección.
- Seleccionar modelos de Deep Learning para la tarea de detección de posibles exoplanetas sobre los datos seleccionados.
- Evaluar resultados de los modelos en la tarea de detección.

CAPÍTULO 2

MARCO CONCEPTUAL

2.1. DETECCIÓN DE EXOPLANETAS

2.1.1. MÉTODO DEL TRÁNSITO

Para enfrentar el problema de detección de exoplanetas se empieza definiendo el fenómeno y su naturaleza que los hace difíciles de encontrar: un exoplaneta es un planeta que orbita una estrella distinta a nuestro sol y, al igual que los planetas de nuestro sistema solar, su luz reflejada es mucho menor en comparación a la luz que emite la estrella que lo ilumina. Es por esto por lo que en la mayoría de exoplanetas su luz reflejada es demasiado tenue para su detección a través de su luz directa [Lang, 2013]. En este contexto han surgido las técnicas de detección indirecta de exoplanetas, entre los cuales se encuentra la prolífica técnica o método del tránsito, la cual considera variaciones temporales en el *flux* del sistema estudiando. El *flux* se define como la cantidad total de energía que cruza una unidad de área por unidad de tiempo, y está dada por la Ecuación (2.1), donde L es la luminosidad¹ de la estrella fuente y r es la distancia entre el observador y la estrella fuente [D. Andrešič, 2021].

$$F = \frac{L}{4\pi r^2} \quad (2.1)$$

La técnica del tránsito detecta un posible exoplaneta considerando que su presencia acompañando una estrella genera variaciones temporales del *flux* del sistema respecto a un sistema sin el exoplaneta. Para detectar este fenómeno debe ocurrir una alineación entre el paso del exoplaneta frente a su estrella y el instrumento que lo observa [Wright y Gaudi, 2012], lo que se conoce como tránsito. Esta variación temporal es detectable a través de atenuaciones periódicas de la señal detectada. En este contexto, se estudian los datos desde observaciones secuenciales a través del tiempo, lo que se conoce como *Time Series* o series de tiempo. En Astronomía es común el estudio de estas series a través de las curvas de luz consistentes en valores de *flux* o magnitud de un objeto astronómico en cierto periodo de tiempo regular o irregular.

Los fotones que entran en la apertura de un telescopio viajan por un conjunto de lentes y/o espejos hasta concentrarse en píxeles CCD de una cámara digital. En este proceso no todos los fotones son detectados debido a que los elementos ópticos nunca transmiten o reflejan el 100 % de la luz, ni los píxeles CCD tienen una eficiencia del 100 % para detectar los fotones. Los fotones que si logran interactuar en la cámara liberan un electrón que es capturado den-

¹La luminosidad de un objeto es una medida de su brillo intrínseco, definida como la energía total emitida en un tiempo fijo.

tro del pozo de potencial del píxel, luego los electrones se convierten con un conteo digital o ADUs (Analogue-to-Digital conversion Units) [AstroLab, 2003]. Tras almacenar esta señal por segundo, se descargan los datos para aplicar técnicas de procesamiento para obtener una curva de *flux* o magnitud que permita su posterior análisis científico. En la Figura 2.1 se muestra un ejemplo de curva de luz generada por el pipeline de procesamiento de datos de la misión espacial TESS (*Transiting Exoplanet Survey Satellite*), donde PDCSAP corresponden a valores de flux nominalmente corregidos para variaciones instrumentales, medido en electrones/segundo.

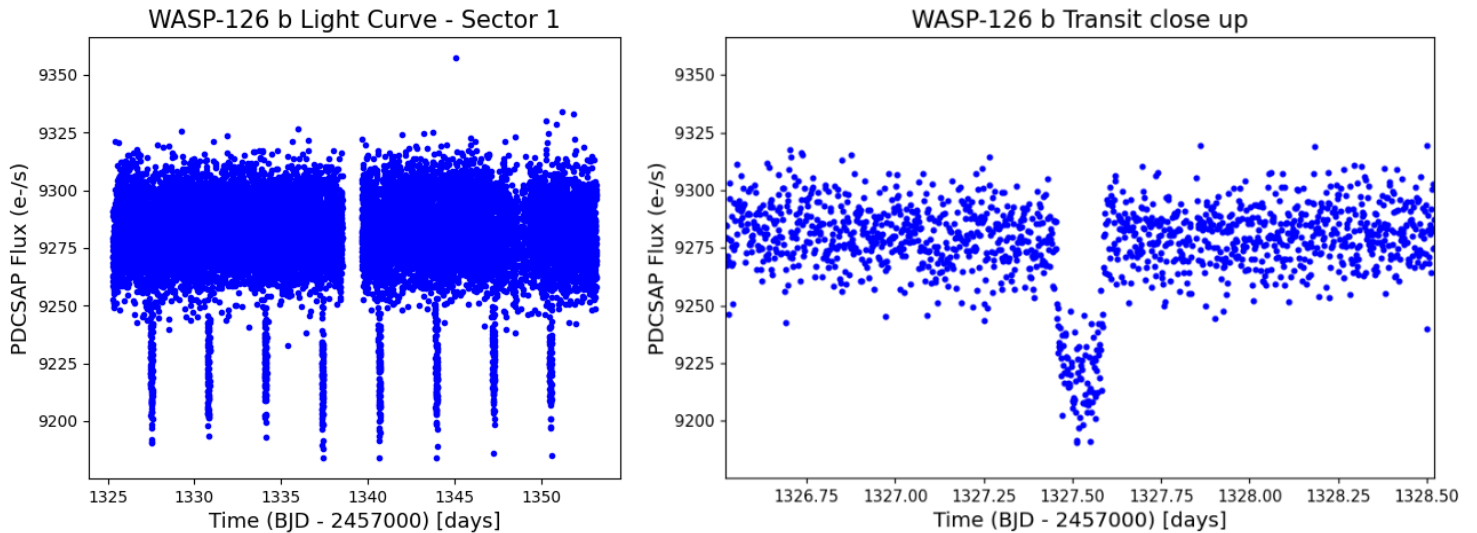


Figura 2.1: (izquierda) una curva de luz del sistema WASP-126 por la misión TESS. Los puntos de PDCSAP Flux que descienden periódicamente corresponden al tránsito del exoplaneta WASP 126-b. (derecha) acercamiento del tránsito centrado en el tiempo 1327.520678 [days]. Fuente: Scott W. Fleming, STScI: https://spacetelescope.github.io/notebooks/notebooks/MAST/TESS/beginner_how_to_use_lc/beginner_how_to_use_lc.html

2.1.2. ANÁLISIS DE CURVAS DE LUZ

Entre las dificultades de la detección de exoplanetas en curvas de luz se encuentra la variedad de clases estelares con luminosidad variable en el tiempo, cuyas curvas pueden ser confundibles con tránsitos exoplanetarios. Ejemplos de estas clases son binarias eclipsantes y estrellas variables. A esto hay que sumar la presencia de ruido en las señales y errores instrumentales producidos durante el apuntamiento del objetivo o fallos en las cámaras producto de las condiciones espaciales. En este escenario, descubrir el periodo del tránsito y una señal característica del fenómeno es esencial para la detección de exoplanetas.

La búsqueda de periodos ocultos en datos de series de tiempo se encuentra dentro del estudio de *Spectral Analysis*, el cual provee herramientas para descubrir periodicidades de señales de distintos fenómenos acústicos, comunicacionales, biomédicos, científicos, etc.

[Cryer y Chan, 2008]. Dependiendo de las características de los datos existen múltiples técnicas para analizar series de tiempos. En este contexto las características son resultado de la adquisición de datos por parte de un instrumento a bordo de un satélite, donde se espera heterocedasticidad en los datos.

Los autores del libro *Modern Statistical Methods for Astronomy With R Applications* lo describen de la siguiente forma: *“Algunos estudios observacionales en astronomía producen variables aleatorias i.i.d. (...) Pero las condiciones i.i.d pueden no cumplirse. La muestra puede ser heterogénea con objetos extraídos de diferentes distribuciones subyacentes. Las observaciones pueden haber sido tomadas bajo diferentes condiciones de modo que los errores de medición difieren. Esto lleva a una condición llamada heterocedasticidad.”* [Feigelson y Babu, 2012]. Respecto a las observaciones, los autores precisan lo siguiente:

1. Para observaciones espaciales, estas se encuentran reguladas por las órbitas de los satélites y los tiempos de asignación de telescopio, los cuales son limitados para adquirir suficientes datos optimizados para su análisis estadístico posterior. Ambos factores hacen que los datos sean adquiridos en tiempos irregularmente espaciados.
2. Las mediciones individuales están sujetas a errores heterocedásticos, donde la varianza de los errores se obtienen desde mediciones independientes de condiciones instrumentales como la relación señal/ruido y calibración.
3. Variaciones de las fuentes astronómicas y el ruido de fondo que debe ser sustraído suelen presentar estructura de correlación.

Los datos irregularmente espaciados, errores instrumentales heterocedásticos y comportamiento temporal cambiante dificultan el uso de técnicas clásicas de análisis espectral como el Análisis de Fourier. Afortunadamente se han desarrollado diversos métodos en astronomía para enfrentar estas dificultades.

Una herramienta útil del Análisis de Fourier clásico son los periodogramas, los cuales son valores que dependen de parámetros ajustados sobre una señal discreta y que son proporcionales a la suma de cuadrados de los coeficientes de la regresión asociados con las frecuencias que componen la señal, donde *“la altura del periodograma muestra la fuerza relativa de los pares coseno-seno en varias frecuencias en el comportamiento general de la serie.”* [Cryer y Chan, 2008]. Los periodogramas se han aplicado en astronomía sobre curvas de luz de estrellas, por ejemplo, en la Figura 2.2 se observa una curva de luz de una estrella variable durante 600 noches consecutivas y en la Figura 2.3 los periodogramas calculados sobre la curva que muestran dos alturas significativas en dos frecuencias.

El análisis clásico de periodogramas requiere datos regularmente espaciados, por lo que en astronomía se han aplicado técnicas para enfrentar esta limitante, por ejemplo, el periodograma Lomb-Scargle (LSP) es una generalización formulada como una modificación del Análisis de Fourier o una regresión de mínimos cuadrados que permite encontrar periodos

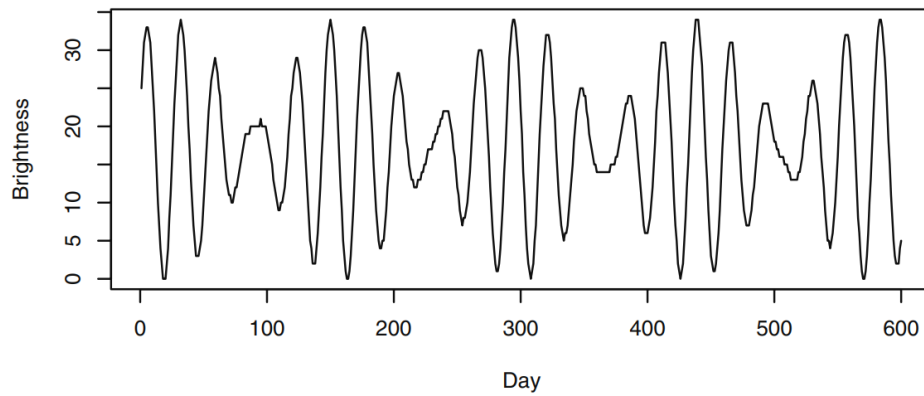


Figura 2.2: Curva de luz de una estrella variable durante 600 noches consecutivas. Fuente: [Cryer y Chan, 2008]

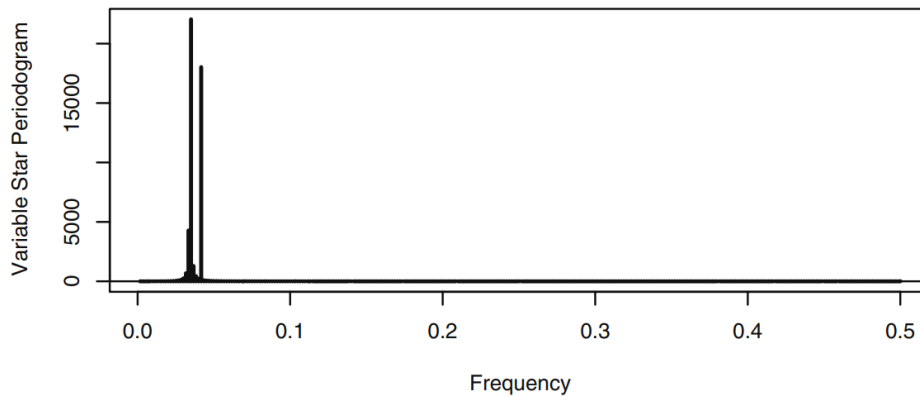


Figura 2.3: Periodogramas calculados sobre la curva de luz de la estrella variable. Fuente: [Cryer y Chan, 2008]

en datos irregularmente espaciados [Feigelson y Babu, 2012]. El uso de periodogramas en curvas de luz ha sido reportado por [D. Andrešič, 2021] para pre-procesar datos de curvas de luz y clasificar tipos de estrellas variables.

Finalmente, al trabajar con curvas de luz hay que tener en cuenta las tendencias (*trends*). *"Incluso con procesamiento post-instrumental, las caídas del tránsito a menudo se ven eclipsadas por tendencias estelares. El ruido estelar es muy específico de una estrella y ocurre con diversas características en todas las escalas temporales. Ciclos de actividad estelar, como el ciclo de actividad solar de 11 años (...) Otras estrellas exhiben variaciones de brillo en muchas escalas temporales diferentes y con un rango de amplitudes"* [Hippke et al., 2019]. Las técnicas utilizadas para remover el *trend* se conocen como *detrending*, con distintos métodos según el contexto de aplicación. En la Figura 2.4 se muestra un ejemplo de *detrending*.

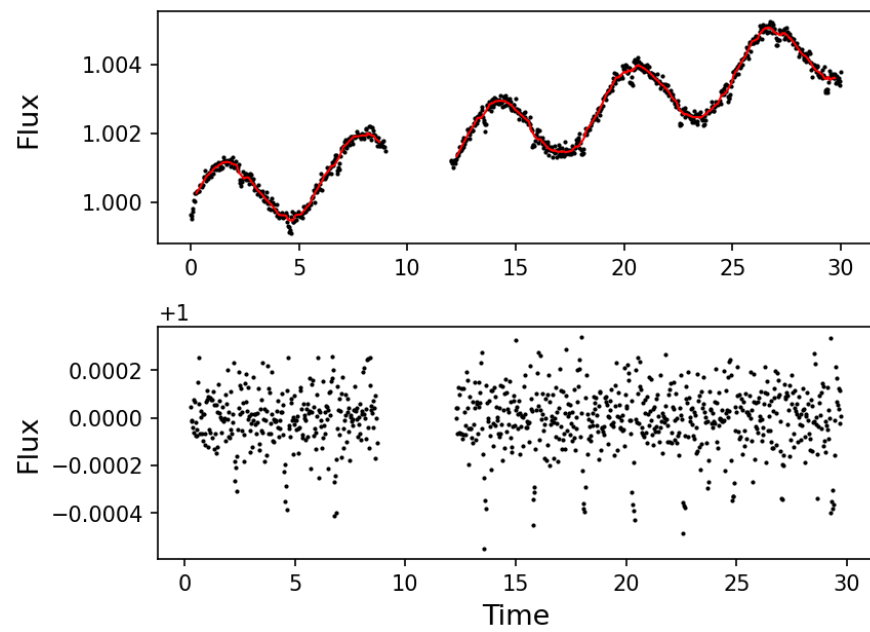


Figura 2.4: Ejemplo de *detrending*. (superior) Curva original. (inferior) *Detrending* aplicado sobre la curva con método *biweight*. Fuente: [Hippke *et al.*, 2019]

2.1.3. REPRESENTACIÓN DE CURVAS DE LUZ DE TRÁNSITOS

La representación de una curva de luz es importante para su efectivo uso como entrada de un modelo de detección. Una técnica muy utilizada en curvas de luz es el *folding time series*, que consiste en plegar los datos en un periodo de prueba o *trial period*, y aplicar métodos estadísticos sobre los datos plegados como *flux* o magnitud en función de la fase.

Asumiendo que la fase se mide en ciclos que empiezan en 0 y terminan en 1, se puede calcular la fase ϕ en función del tiempo t según la ecuación (2.2), donde t_0 es el tiempo donde inicia el ciclo (llamado *epoch*) y p es un periodo dado. De esta forma, $\phi \in [0, 1]$. Graficando la curva de luz, ahora como *flux* en función de fase (en vez de *flux* en función de tiempo) se obtiene una curva de luz *phase-folded* o *phase diagram* [AAVSO, 2010].

$$\phi = \lfloor \frac{t - t_0}{p} \rfloor = (\frac{t - t_0}{p})_{mod 1} \quad (2.2)$$

Una utilidad de este método es que si el periodo de prueba corresponde con una periodicidad de un patrón presente en la curva o es cercano a ella y el *epoch* es correcto, la curva *phase-folded* con $\phi \in [-1, 1]$ (equivalente a concatenar el ciclo previo) puede revelar dicho patrón centrado en la mitad. En este contexto, el patrón de interés es el tránsito exoplanetario, similar al presentado en la Figura 2.5 (izquierda), cuyo periodo verdadero corresponde a 3.288776 [days]. En la curva derecha se puede observar que usando un periodo incorrecto se puede perder la señal característica correspondiente al tránsito.

Algunos métodos que utilizan el *phase-folding* son apropiados para variaciones debido a eclipses orbitales o rotacionales con la forma característica *flat-bottom U-shape*, además son insensibles a la duración y espaciado irregular de los datos y permiten errores de medición heterocedásticos [Feigelson y Babu, 2012].

En la búsqueda de exoplanetas en curvas de la misión espacial Kepler, Shallue y Vanderburg [Shallue y Vanderburg, 2018] utilizan curvas *phase-folded* centradas en eventos TCE (*Threshold Crossing Event*). Estos eventos corresponden a secuencias presentes en las curvas de luz que se asemejan a la señal de un exoplaneta en tránsito con un grado suficiente para justificar un análisis posterior [Christiansen, 2012], pudiendo ser un tránsito real o un falso positivo. Estas curvas son utilizadas para generar dos representaciones de curvas de luz al dividir las en intervalos (*bins*) y formar vectores 1D según el siguiente procedimiento:

1. Definir una secuencia uniforme de intervalos (*bins*) en el eje del tiempo con ancho δ y distancia λ entre los centros de los intervalos.
2. Por cada intervalo, calcular la mediana de *flux* de los puntos que caen dentro del intervalo. El resultado es un vector 1D por curva (input del modelo de detección).

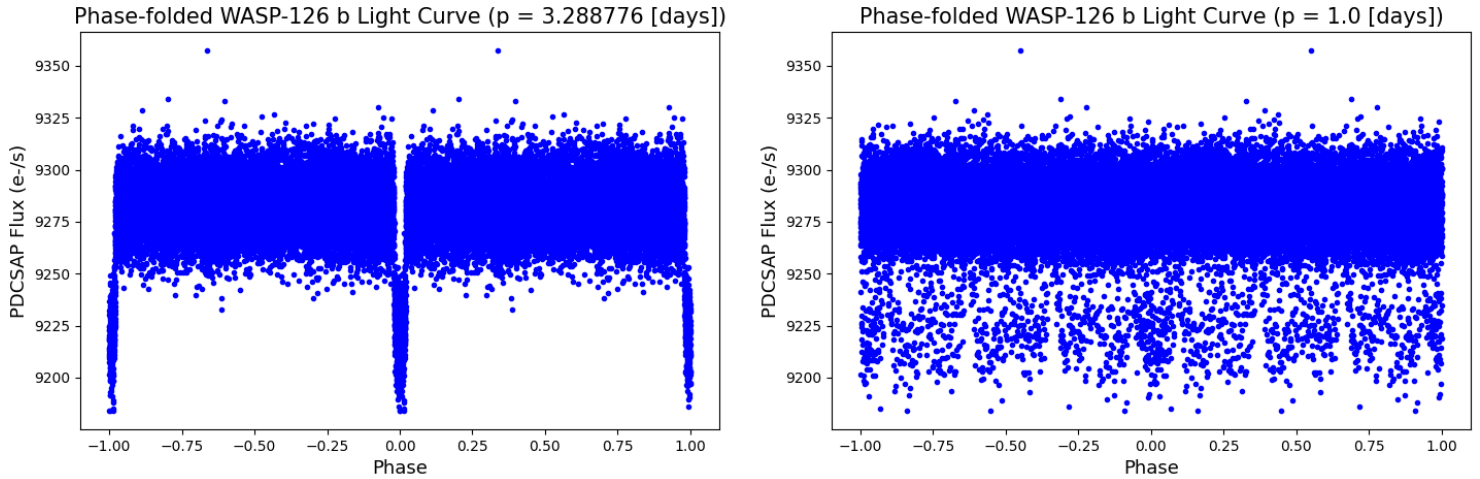


Figura 2.5: (izquierda) curva *phase-folded* del sistema WASP-126 utilizando el periodo de tránsito del exoplaneta WASP-126 b, usando los mismos datos de la Figura 2.1. (derecha) curva *phase-folded* usando un periodo distinto del tránsito. El valor de epoch es el mismo para ambas curvas (1327.520678 [days]). Fuente: Elaboración propia.

Dependiendo de los valores de δ y λ se producen distintas representaciones. Si $\delta = \lambda$ cada dato cae exactamente en un intervalo. Por otro lado, $\delta > \lambda$ presenta superposición de datos en los intervalos. Los autores utilizan dos valores de λ para generar dos vistas (*view*) de la curva para usar como input del modelo de detección. La primera de ellas, la **global view**, considera un valor de λ como una fracción del periodo del TCE, lo que genera una representación de tamaño fijo de la curva entera. La segunda vista, la **local view**, considera un valor de λ como una fracción de la duración del TCE, lo que genera una representación de tamaño fijo de una ventana alrededor del tránsito. Ambas representaciones son utilizadas como entrada para entrenar una red neuronal con capas convolucionales. En la Figura 2.6 se muestran ambas vistas para una curva procesada por [Yu *et al.*, 2019] proveniente de la misión espacial TESS, siendo ambas representaciones 1D de largo 201 y 61 respectivamente.

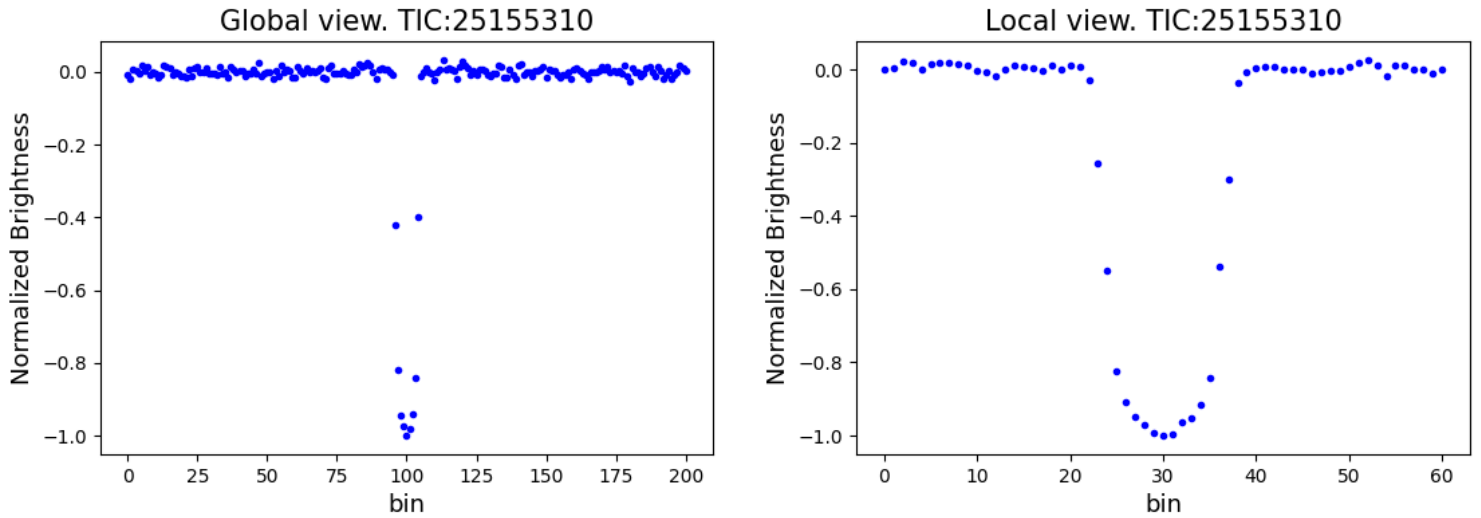


Figura 2.6: Vista global y local de la curva *phase-folded* centrada en un TCE del objetivo TESS Input Catalog (TIC) 25155310. En esta curva hay un exoplaneta confirmado. Fuente: elaboración propia utilizando los datos ya procesados de [Yu *et al.*, 2019].

Por último, para generar curvas de luz con tránsito de forma sintética es necesario un modelo que considere las características de la estrella, del exoplaneta y su órbita respectiva. Un aspecto relevante de las estrellas es el oscurecimiento en el borde o *limb darkening*. Este fenómeno es apreciable en nuestro propio sol tal como se muestra en la Figura 2.7, donde la luz del borde del disco solar es menos intensa y rojiza que el centro. La razón de este oscurecimiento es el gradiente de temperatura de la fotosfera solar, que hace que los rayos de luz que escapan del sol cerca del borde tengan menor temperatura respecto a los rayos que escapan del centro.

El autor [Claret, 2003] reafirma que el *limb darkening* no es un fenómeno lineal y recopila leyes de su modelado con comportamiento lineal, cuadrático, raíz cuadrada y logarítmico. Los comportamientos lineal y cuadrático se expresan con las ecuaciones (2.3) y (2.4), respectivamente, donde $I(1)$ es la intensidad específica en el centro del disco estelar, a y b son coeficientes y el ángulo γ es entre la línea de visión y el emergente de intensidad². Luego presenta una nueva ley de *limb darkening* basada en el método de mínimos cuadrados con 4 coeficientes que se expresa en la ecuación (2.5).

La ley cuadrática anteriormente presentada es utilizada por los autores Mandel y Agol para proponer una fórmula analítica para el eclipse de una estrella por un tránsito exoplanetario explicando que el efecto de *limb darkening* es significativo durante un eclipse. [Mandel y Agol, 2002]. Un ejemplo de curva de luz computada por este algoritmo se muestra en la Figura 2.8, cuyos parámetros son los mismos usados por [Aigrain y Favata, 2002] en su propuesta de método bayesiano para la detección de periodos. La simulación considera un planeta tipo Júpiter con una órbita corta alrededor de una estrella similar al sol con las

²Ángulo entre la normal a la superficie estelar y la línea de visión al observador [Mandel y Agol, 2002]

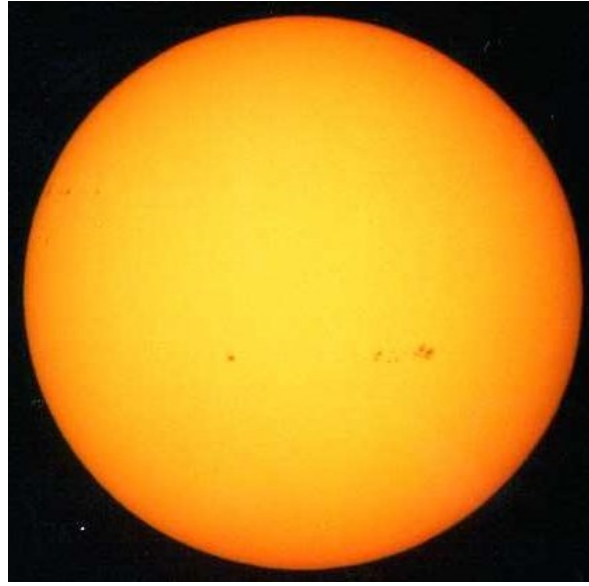


Figura 2.7: Efecto de *limb darkening* en nuestro sol. Fuente: <http://spiff.rit.edu/classes/phys440/lectures/limb/limb.html>.

características presentes en la Tabla 2.1.

$$\mu = \cos(\gamma), \quad \frac{I(\mu)}{I(1)} = 1 - \mu(1 - \mu) \quad (2.3)$$

$$\frac{I(\mu)}{I(1)} = 1 - a(1 - \mu) - b(1 - \mu)^2 \quad (2.4)$$

$$\frac{I(\mu)}{I(1)} = 1 - a_1(1 - \mu^{\frac{1}{2}}) - a_2(1 - \mu) - a_3(1 - \mu^{\frac{3}{2}}) - a_4(1 - \mu^2) \quad (2.5)$$

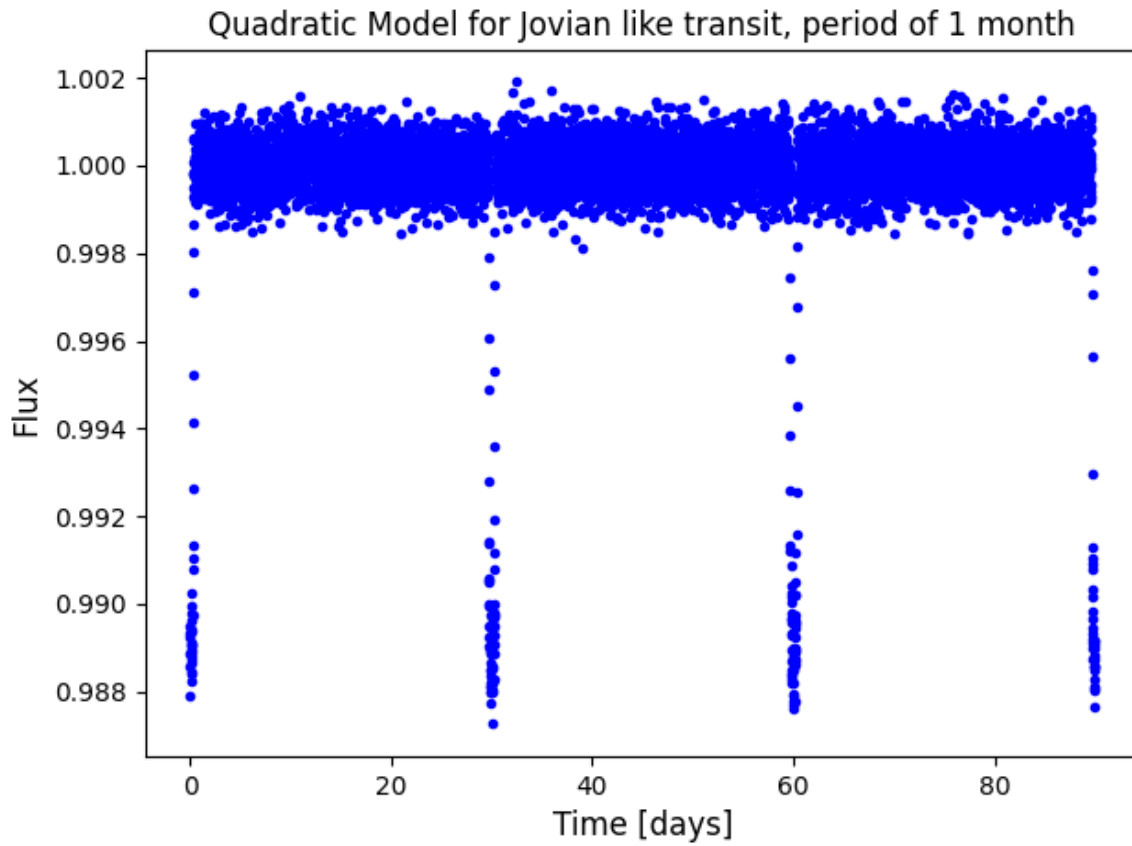


Figura 2.8: Simulación de curva de luz usando el algoritmo de Mandel y Agol. A la señal se le aplica ruido gaussiano con media cero y desviación estándar 1. Fuente: elaboración propia.

Característica de entrada	Valor
Time step	15 minutos
Radio estelar (R_{sun})	1 (Radio solar)
Luminosidad estelar (L_{sun})	1 (Luminosidad solar)
Radio planetario (R_{jup})	1 (Radio júpiter)
Periodo de tránsito (P)	2880 minutos (1 mes)
Excentricidad (e)	1
Argumento del periastro	0
Inclinación	$0,5\pi [rad]$
Duración total de curva	$3P$
Distancia estrella-planeta	$15,3R_{sun}$

Tabla 2.1: Parámetros de entrada del algoritmo Mandel y Agol para producir la curva de la Figura 2.8. Fuente: elaboración propia.

2.2. MISIONES DE BÚSQUEDA DE EXOPLANETAS

2.2.1. MISIONES TRADICIONALES

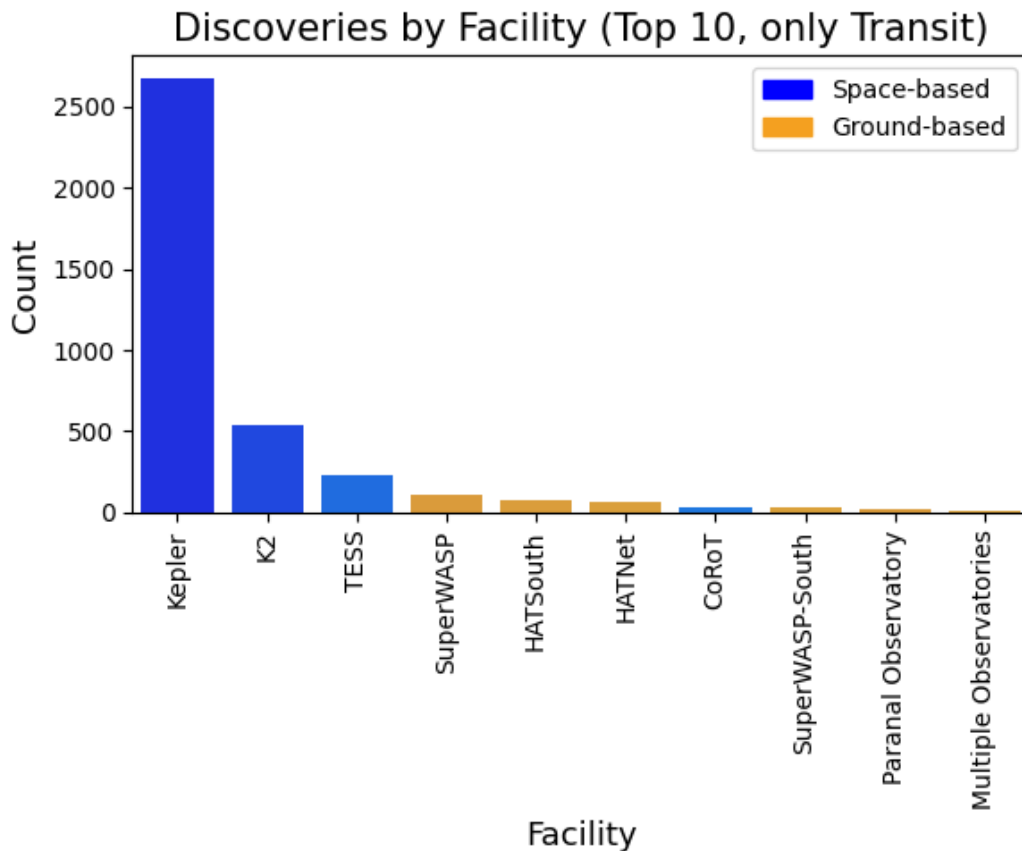


Figura 2.9: Descubrimientos de exoplanetas confirmados por misión/instalación usando el método del tránsito (23/07/2022)). Fuente: elaboración propia usando datos de NASA Exoplanet Archive.

Distintos telescopios de gran envergadura han sido lanzados al espacio para realizar fotometría estelar en la búsqueda de exoplanetas. Pese a que los instrumentos a bordo de estos satélites actualmente superan la precisión de un instrumento a bordo de un nanosatélite, al igual que son capaces de generar mayor cantidad de datos, es de interés estudiar estas misiones por la amplia literatura existente en relación a sus *pipelines* de procesamiento de datos y modelos de detección. En la Figura 2.9 se presentan las 10 misiones/instalaciones con mayor número de exoplanetas descubiertos usando el método de tránsito. A continuación se describen los 3 proyectos espaciales con mayor número de descubrimientos.

La principal misión en la búsqueda de exoplanetas por número de exoplanetas descubiertos y confirmados (2680 exoplanetas a la fecha 23/07/2022) es la **Kepler Mission** o **Kepler**, cuyo lanzamiento fue el 9 de marzo de 2009 en una órbita heliocéntrica (periodo de 372.5 días).

Esta fue diseñada "para determinar la frecuencia de exoplanetas del tamaño de la Tierra o mayor en la zona habitable³ en estrellas similares al sol y para caracterizar la diversidad de sistemas planetarios" [Borucki, 2016].

El campo de visión (*field of view*, FoV) de la misión Kepler es fijo, cercano al ecuador galáctico, rico en número de estrellas y lejos de estrellas muy brillantes que saturan sus detectores. El FoV seleccionado permite la observación de 170.000 estrellas simultáneamente.

La selección de estrellas objetivos para el proyecto resulta en la creación del *Kepler Input Catalog* o KIC, formado principalmente por estrellas enanas en secuencia principal lo suficientemente brillantes y pequeñas para presentar tránsitos detectables de exoplanetas pequeños en la zona habitable.

Los datos científicos de las 170.000 estrellas son recolectados en un tiempo de integración de 6.02 segundos con tiempos de lectura⁴ de 0.52 segundos, luego, los datos están disponibles en dos cadencias: la *Long Cadence* (LC) integra 270 exposiciones para generar 29.4 minutos de observación, mientras que en la *Short Cadence* (SC) se integra 9 exposiciones, lo que equivale a un punto de dato cada 58.9 segundos. [Murphy, 2012].

Los datos descargados del satélite se convierten en *fluxes* usando un *pipeline* que calibra los datos píxeles, realiza *aperture photometry* y corrige errores sistemáticos. Los datos generados son almacenados públicamente en MAST (Mikulski Archive for Space Telescopes), y pueden ser descargado desde su portal⁵, sitio que aloja curvas de luz LC, SC y *target pixel files*. Adicionalmente se pueden acceder a los datos desde NASA Exoplanet Archive⁶. El identificador utilizado en el KIC es *kepid* y el identificador usado para los objetos de interés o Kepler Object of Interest (KOI) es *koi_name*, los cuales son objetivos identificados con una secuencia similar a un tránsito.

En mayo de 2013, la habilidad de Kepler de monitorizar un gran campo con alta precisión es comprometida debido a un fallo en la segunda rueda de reacción (de un total de cuatro). En consecuencia se ejecutó una misión extendida llamada **K2** con menor precisión fotométrica [Vanderburg y Johnson, 2014]. Esta extensión fue prolífica con 537 exoplanetas confirmados a la fecha 23/07/2022.

Siguiendo a Kepler y K2 en número de exoplanetas descubiertos y por confirmar se encuentra el **Transiting Exoplanet Survey Satellite** o **TESS**. El diseño de la misión previo a su lanzamiento es descrito por [Ricker *et al.*, 2016] con las siguientes características:

Su objetivo primario es descubrir cientos de tránsitos de planetas de tamaño menor a Neptuno y cuyas estrellas son lo suficientemente brillantes como para realizar estudios espec-

³La zona habitable es el área alrededor de una estrella donde el agua líquida puede existir en la superficie de un planeta orbitando.

⁴Tiempo requerido para digitalizar.

⁵<https://archive.stsci.edu/>

⁶<https://exoplanetarchive.ipac.caltech.edu/>

troscópicos que permitan conocer la masa y composición atmosférica del exoplaneta. Para ello se diseñó un *survey* de 2 años de todo el cielo (*all-sky*), considerando periodos orbitales desde las pocas horas cercanas al límite de Roche, hasta un año o más, considerando un bias de periodo con un máximo de 10 [days] tomando en cuenta la eficiencia de detección y ocurrencia de tránsitos.

El diseño considera una órbita cercana a la tierra que ofrezca observaciones por semanas con perturbaciones de actitud que permitan precisión fotométrica estable, sumado a una estabilidad frente a la radiación para reducir la degradación de las cámaras CCDs e instrumentación, además de facilitar la telemetría. Dicha órbita fue determinada como elíptica alrededor de la Tierra, con un periodo orbital de 13.7 [days].

La carga útil del satélite o payload incluye cuatro cámaras idénticas, cada una con un lente ensamblado con siete piezas ópticas y un detector con 4 CCDs, los cuales producen un stream de imágenes con tiempo de exposición de 2 segundos, los cuales pueden sumarse para obtener una exposición efectiva de 2 minutos. Las 4 cámaras funcionan como un array que provee un campo de visión (FoV) de $24^\circ \times 96^\circ$, lo que permite observar un hemisferio celeste en un año en múltiples sectores. El detalle de las características de los lentes se encuentran en la Tabla 2.2.

El principal producto son colecciones de *subarrays*, típicamente de 10 x 10 pixeles, centradas en estrellas objetivo preseleccionadas. Estos productos son los que posteriormente se post-procesan en la Tierra. Esta selección de estrellas incluye un catálogo ideal de 200.000 estrellas en secuencia principal que son lo suficientemente brillantes y pequeñas para permitir la detección de exoplanetas más pequeños que Neptuno. El pipeline de procesamiento de datos está basado en el desarrollado para la misión Kepler, lo que incluye calibración a nivel de *pixel*, *background subtraction*, *aperture photometry*, *detrending* y búsqueda de tránsitos.

Field of view of each lens	$24^\circ \times 24^\circ$
Combined field of view	$24^\circ \times 96^\circ$
Entrance pupil diameter	10.5 [cm]
Focal ratio	f/1.46
Wavelength range	600-1000 [nm]

Tabla 2.2: características de los lentes de TESS. Fuente: [Ricker *et al.*, 2016]

Adicionalmente a las observaciones cada 2 minutos, la misión contempla observaciones *full frame images* con una cadencia efectiva de 30 minutos, las cuales capturan todo el campo de visión. Una vez cumplidos los dos años de misión primaria, se espera una misión extendida.

De forma análoga a Kepler, el TESS Input Catalog o TIC es usado para seleccionar los objetivos de la misión y calcular las propiedades físicas y observacionales de los candidatos a exoplaneta. Su versión 8 utiliza el catalogo GAIA DR2 como base y fusiona un gran número de otros catálogos fotométricos tales como 2MASS, UCAC4, APASS, SDSS, WISE, etc.

[de Données astronomiques de Strasbourg, 2022]. Adicionalmente a este catálogo existe el Candidate Target List o CTL, cuyo propósito es *"proporcionar un subconjunto de objetos TIC para seleccionar las estrellas objetivo para observaciones de cadencia de 2 minutos en servicio de los requisitos científicos primarios de TESS"* [Stassun et al., 2018].

La misión TESS fue lanzada al espacio el 18 de abril de 2018 e inició sus operaciones científicas el 25 de julio del mismo año. Ya finalizada su misión primaria, ha observado decenas de millones de estrellas [Kunimoto et al., 2022] , con 6025 exoplanetas confirmados o por confirmar (20/07/2022). Los datos públicos de TESS, al igual que la misión Kepler, se encuentran en MAST y NASA Exoplanet Archive.

2.2.2. ASTRONOMÍA CON NANOSATÉLITES

El profesor de astrofísica Evgenya Shkolnik [Shkolnik, 2018], Serjeant, S et al. [Serjeant et al., 2020] y Douglas E. [Douglas et al., 2019] exponen un estado del arte las diversas posibilidades del uso de nanosatélites para estudiar fenómenos astronómicos. Describen las misiones pasadas, existentes y futuras especificando los objetivos, oportunidades tecnológicas y espectro electromagnético a estudiar. A grandes rasgos las investigaciones se centran en fotometría estelar, espectroscopia, estudio en rayos X, radio, estudio solar, GRB, clúster galácticos y demostraciones tecnológicas. Con sus descripciones e investigación propia se resumen las misiones más relevantes para los datos a trabajar en esta memoria (Tabla 2.3). Cabe destacar la misión BRITe, la cual consiste en 5 nanosatélites que observan estrellas brillantes. Los datos de esta misión son relevantes ya que pueden contener posibles tránsitos exoplanetarios. Un esquema de un nanosatélite BRITe es presentado en la Figura 2.10.

Misión	Estudio	Objetivo	Estado	Datos disponibles
BRITE	Fotometría estelar, estrellas brillantes.	Diferencia de fotometría con millimagnitud (0.1 % error) de estrellas brillantes con 5 nanosatélites (6 originalmente).	Lanzado en 2014	Sí
ASTERIA	Fotometría estelar, tránsitos, demostración tecnológica.	Medir tránsito exoplanetarios con fotometría <100 partes por millón.	Lanzado en 2017	Sí
CUTE	Espectroscopía, exoplanetas.	Espectroscopia de tránsito de exoplanetas en el UV cercano.	Lanzado en 2021	Aún no públicos
SPARCS	Fotometría estelar, estrellas de baja masa, exoplanetas	Estudio UV de estrellas de baja masa, con posibles exoplanetas.	Futuro (2023)	No
CubeSpec	Prueba de concepto y demostración. Espectroscopia de estrellas masivas pulsantes	Obtener datos espectroscópicos ópticos de alta cadencia y resolución para facilitar la astrosismología de estrellas masivas pulsantes [Bowman <i>et al.</i> , 2022].	Futuro (2024)	No

Tabla 2.3: Tabla de misiones con nanosatélites de interés dentro del trabajo de investigación.
Fuente: elaboración propia.

2.2.3. MISIÓN BRITE

Las curvas de luz producidas por este proyecto son la principal fuente de datos de esta memoria, lo que incluye curvas descargables para 609 estrellas diferentes a la fecha (26/07/2022). A continuación se resume el proyecto según lo descrito en [Weiss *et al.*, 2014]:

La concepción del proyecto es resultado de la colaboración internacional de Austria, Canadá y Polonia. El diseño concluyó en una constelación de 3 pares de nanosatélites capaces de observar en bandas espectrales rojas ($550 - 700[nm]$) y azules ($390-460[nm]$), los cuales son operados por los 3 países. Los detalles generales de los 6 nanosatélites están presentes en la Tabla 2.4.

El objetivo científico del proyecto es observar gran parte del cielo, midiendo variaciones de brillo y temperatura de estrellas brillantes (típicamente $V < 5[mag]$) en tiempos desde pocos minutos hasta varios meses o años mediante fotometría de doble banda. La motivación de estudiar estrellas brillantes es complementar las curvas de luz con una mayor precisión respecto a telescopios terrestres debido a la falta de estrellas de comparación lo suficientemente cercanas a los objetivos, lo cual es necesario para realizar fotometría diferencial.

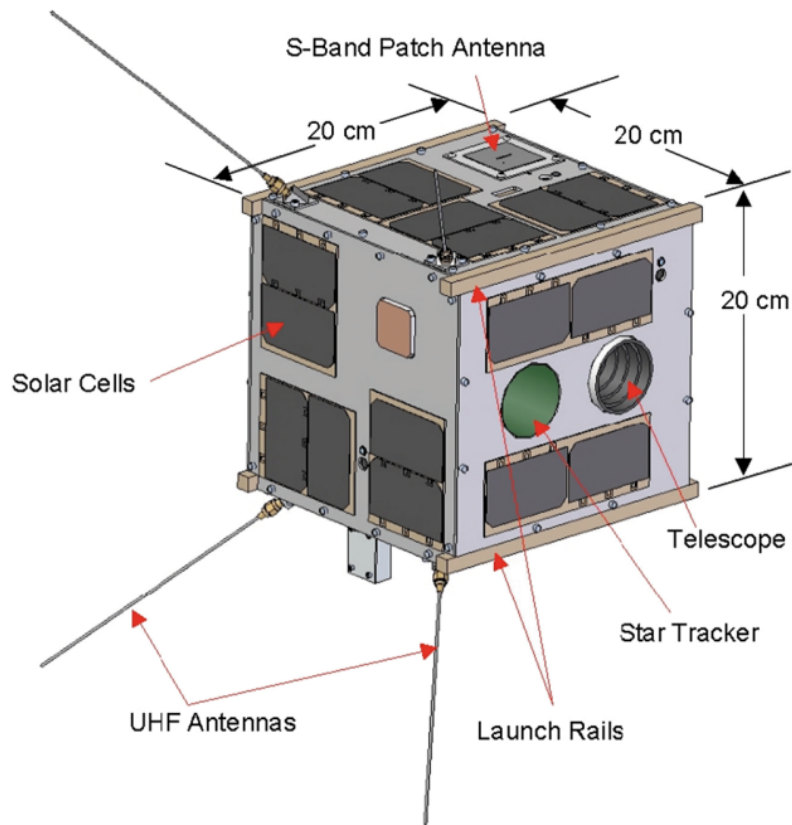


Figura 2.10: Esquema de un nanosatélite BRITE, utilizado para fotometría de estrellas brillantes. Fuente: [Weiss *et al.*, 2014]

Esta constelación tiene el potencial de observar cualquier dirección del cielo, con objetivos generalmente ocultos por la Tierra durante parte de las observaciones. Esto produce separaciones periódicas en los datos adquiridos, sin embargo, las separaciones se pueden mitigar con el resto de nanosatélites al poseer todos diferentes órbitas.

Originalmente se proponen 3 formas diferentes en que la fotometría es recolectada e inicialmente procesada:

1. Fotometría *subraster*: Hasta 15 *subrasters* desde lecturas típicas cada pocos segundos. Cada *subraster* consiste en 32 x 32 píxeles centrados en la estrella objetivo, con una posible reducción a 24 x 24 píxeles. Estos *raster* pueden sumarse a bordo antes de ser descargados, lo que resulta en un tiempo de integración desde pocos minutos hasta 45 minutos por órbita. En la práctica los *subraster* son cuadrados o rectangulares, con rangos que oscilan entre 24 y 54 píxeles según la estrella y el nanosatélite.
2. Procesamiento a bordo que puede ser implementado a pedido que considera hasta 100 *subrasters* (32 x 32 píxeles) cada pocos segundos. Cada uno de los *subraster* incluye parámetros calculados a bordo como la intensidad media de los píxeles en las

ID/filter	Full name	Launch	Inclination [°]	Period [min]
UB ^r	UniBRITE	25/02/2013	98.4	100.3
BA ^b	BRITE-Austria	25/02/2013	98.4	100.3
BL ^b	Lem	21/11/2013	97.7	99.5
BT ^r	BRITE-Toronto	19/06/2014	97.8	98.2
BM ^b	BRITE-Montreal	19/06/2014	-	-
BH ^r	Heweliusz	19/08/2014	97.9	97.0

Tabla 2.4: Resumen de los 6 nanosatélites del proyecto BRITE. BM^b no entró en operaciones debido a una falla en la separación del vehículo de lanzamiento. Fuente: elaboración propia. Datos orbitales de n2yo.com (20/05/2022)

4 esquinas del *subraster* (permitiendo caracterizar el *background*), la suma de valores de píxeles dentro de dos círculos concéntricos centrados en el objetivo (permitiendo fotometría de apertura) y valores estadísticos como medianas y desviaciones estándar.

3. *Full frames* capturados al inicio de las observaciones de cada campo. Consisten en una imagen CCD completa de 11 MP cuyo propósito es detectar cambios en los CCD como píxeles calientes resultantes del daño por radiación. Debido al tamaño de estas imágenes, la telemetría requiere más de 3 días para su descarga, por lo que solo se producen pocas veces durante la misión.

Con el objetivo de lograr fotometría de alta precisión y no la producción de imágenes estelares finas, la instrumentación considera un pequeño telescopio basado en lentes y con un detector CCD sin refrigeración. Dicho telescopio considera una apertura de 30 [mm] y largo focal efectivo de 70 [mm], permitiendo un FoV de $24^{\circ} \times 20^{\circ}$. El diseño varía levemente según el color del filtro, y el detector CCD al no ser refrigerado activamente tiene una temperatura según el bus del nanosatélite, con un promedio simulado de 10 a 40 °C.

Con observaciones tempranas durante el comisionamiento revelaron una sensibilidad del CCD mayor a la esperada frente a la radiación, lo que produce más píxeles malos (calientes) de lo esperado. Los problemas técnicos y mitigaciones relacionadas a la radiación son presentados en [Pablo *et al.*, 2016] y se describen a continuación:

La radiación (protones energéticos y electrones atrapados en la magnetósfera terrestre) producen daños acumulables en el tiempo en la electrónica, memoria y detectores CCD. El ruido en una CCD es causado por rayos cósmicos que generan señales falsas al depositar su energía en los píxeles del CCD. Otro daño es el desplazamiento, lo cual ocurre cuando un rayo cósmico desplaza un átomo de silicio en el sustrato del CCD, lo que provoca un daño persis-

tente. Estos daños introducen los llamados píxeles calientes (*hot pixels*) o columnas calientes (*warm columns*), junto a un aumento de la ineficiencia en la transferencia de carga (CTI).

En respuesta a estos problemas, los nanosatélites BMb y BTr fueron equipados con un escudo de radiación adicional antes de su lanzamiento, y para todos los nanosatélites se aplican maniobras de reorientación respecto a sol para reducir variaciones de temperatura en respuesta a la ausencia de mecanismo de refrigeración. Adicionalmente se aplica una estrategia de software efectiva en tiempo llamada “*chopping*” para mitigar el efecto en la fotometría. Esta estrategia se logra al desplazar el telescopio apuntando hacia adelante y hacia atrás en dos exposiciones sucesivas. Esto mantiene la estrella objetivo en los *subrasters* de ambos desplazamientos con la ventaja de que el *point-spread function* (PSF) estelar de las imágenes no se superponen (con una separación de unos 20 píxeles), lo que permite que al restar ambos *frames* se obtenga una imagen con un objetivo PSF positivo y otro negativo, donde los defectos de *background* son mayoritariamente eliminados. En la Figura 2.11 se muestra el efecto de esta técnica.

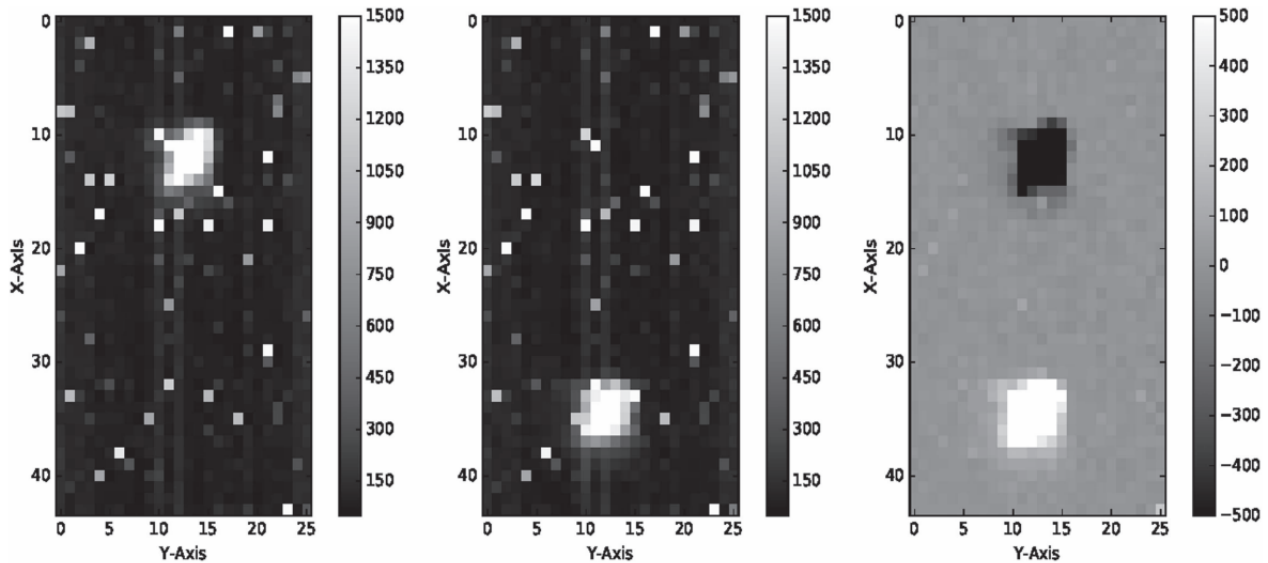


Figura 2.11: Efecto de la técnica “chopping”. Los subraster izquierdo y medio corresponden a exposiciones consecutivas de la estrella HD23630, ambos con un PSF estelar dominante. A la derecha se aprecia la resta de ambos subrasters, donde los defectos de background son removidos. Fuente: [Pablo *et al.*, 2016]

El procesamiento de datos y fotometría es presentado en [Popowicz *et al.*, 2017] y es descrito a continuación:

Los *subrasters* descargados de la constelación son cuadrados para el modo de observación *stare* y rectangulares para el modo “*chopping*”. Para evitar *undersampling* del perfil estelar, lo que implica una fotometría menos precisa, las estrellas en las imágenes de BRITE están fuera de foco.

Durante los primeros dos años desde el primer lanzamiento (UBr y BAb), las observaciones fueron realizadas en el modo “*stare*”, en el cual los *trackers* mantienen el nanosatélite en una orientación fija que permite generar imágenes consecutivas con la estrella localizada aproximadamente en la misma posición cercana al centro de los *subrasters*. Debido a los problemas por la radiación descritos anteriormente, la fotometría del modo “*stare*” fue fuertemente afectada. En respuesta, **el modo “*chopping*” es el modo por defecto utilizado en el proyecto para la recolección de datos hasta la fecha actual.**

2.2.4. MISIÓN ASTERIA

El *Arcsecond Space Telescope Enabling Research In Astrophysics* (ASTERIA) es un nanosatélite CubeSat (6U) diseñado para demostrar tecnologías de alta precisión fotométrica con una plataforma pequeña. Esta misión fue desplegada desde la Estación Espacial Internacional el 20/11/2017, finalizando su misión primaria en 2018 una vez verificadas con éxito las demostraciones tecnológicas requeridas. Posterior a eso la misión realizó una extensión de dos años hasta la pérdida de contacto en 2019 [Knapp *et al.*, 2020].

Entre los objetivos científicos de ASTERIA se encuentra la estrella tipo sol 55 Cancri, la cual posee 5 exoplanetas, de los cuales uno (55 Cancri e) realiza tránsitos con un periodo de 18 horas.

2.3. MODELOS DE DEEP LEARNING

2.3.1. INTRODUCCIÓN

El Aprendizaje Profundo o *Deep Learning* se encuentra dentro de la disciplina de Inteligencia Artificial, en particular forma parte del campo del Aprendizaje Automático o *Machine Learning*, "campo que se ocupa del cómo construir programas computacionales que mejoren automáticamente con la experiencia", donde "un programa aprende de la experiencia E respecto a una clase de tareas T y medida de desempeño P , si el desempeño en las tareas T , medido por P , mejora con la experiencia E [Mitchell, 1997]".

En este contexto, un modelo profundo consiste en múltiples niveles de computación entrenables que implementan transformaciones aplicadas a datos de entrada X para obtener datos de salida Y . El modelo estrella del aprendizaje profundo es la Red Neuronal Artificial o *Artificial Neural Network*(ANN), la cual es un grafo de computación que implementa una función entre dos espacios X e Y .

Una red neuronal puede ser utilizada para resolver problemas de aprendizaje supervisado, los cuales consisten en encontrar automáticamente una función que permita predecir el *output* Y correcto o deseado a partir de un *input* X (ejemplo de entrenamiento). En este tipo de aprendizaje se conoce de antemano la salida asociada a cada ejemplo de entrada. El aprendizaje supervisado es de interés en este trabajo en la tarea de clasificación de ejemplos de entrada en categorías, es decir, dados los atributos que describen un ejemplo X , predecir su categoría Y dentro de un conjunto finito de clases. Dicha categoría se conoce como etiqueta del ejemplo y en el contexto de este trabajo, un ejemplo de categorías utilizadas por [Yu et al., 2019] son Planeta Candidato, Binaria Eclipsante, Variabilidad estelar y Ruido instrumental.

La unidad básica de una red neuronal es la neurona, la cual es un modelo que aplica una función g (típicamente no lineal) a una combinación lineal de los valores de entrada de la neurona y parámetros entrenables de esta misma. En la ecuación (2.6) se presenta este modelo con los datos $x_1, x_2, \dots, x_d$ que recibe la neurona, pesos $w_1, w_2, \dots, w_d$ y un umbral b (llamado bias). Los parámetros w_i y b son entrenables y $g(\xi)$ es el valor de salida de la neurona correspondiente al resultado de aplicar g sobre la preactivación ξ .

$$g(\xi) = g\left(\sum_{i=1}^d w_i x_i - b\right) \quad (2.6)$$

La función g (llamada función de activación) da nombre a distintos tipos de neuronas. La elección más simple es aplicar una función lineal tal como se describe en la ecuación (2.7), sin embargo, al explorar funciones no lineales se presenta mayor utilidad, por ejemplo, la neurona Sigmoidal implementa la ecuación (2.8) y tiene aplicación en escenarios donde se

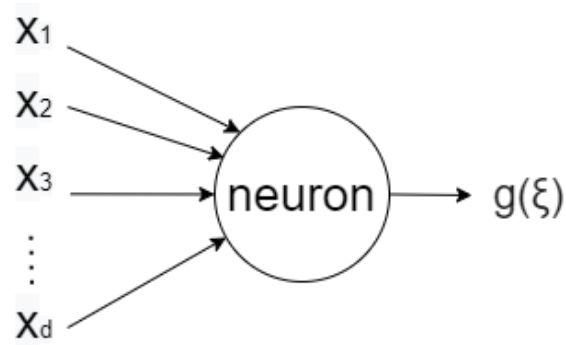


Figura 2.12: Ejemplo de neurona. Fuente: Elaboración propia.

busca predecir la probabilidad de una clase binaria.

$$g(\xi) = \xi \quad (2.7)$$

$$g(\xi) = \frac{1}{1 + e^{-\xi}} \quad (2.8)$$

Otra alternativa popular en Deep Learning es la neurona rectificadora o ReLU que implementa la ecuación (2.9) y que permite reducir la complejidad efectiva de una red.

$$g(\xi) = \max(0, \xi) \quad (2.9)$$

2.3.2. ARQUITECTURAS

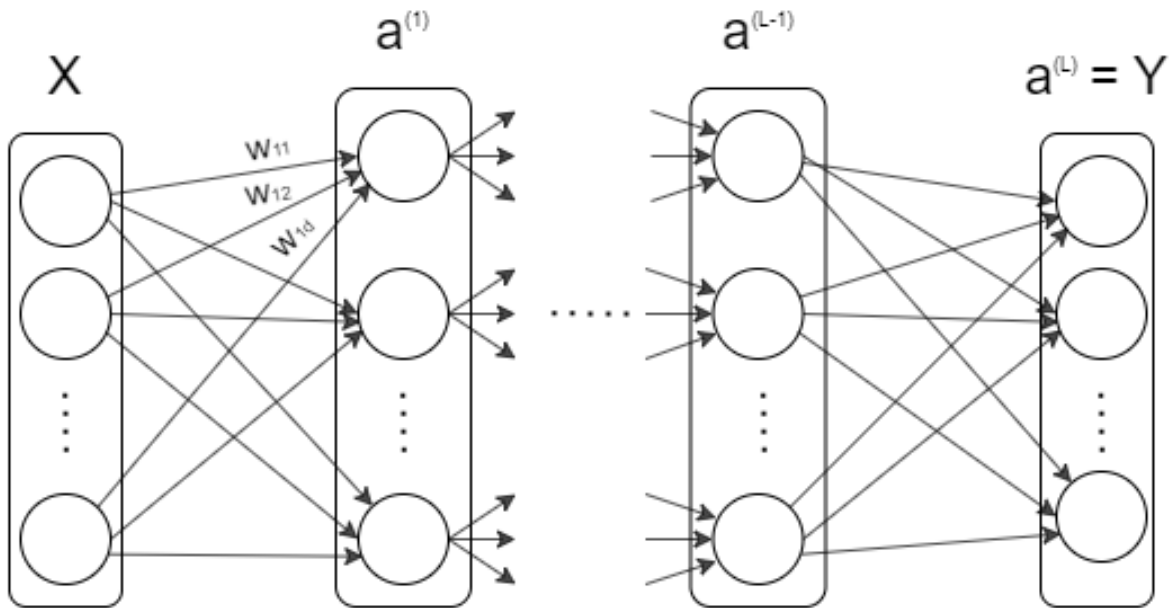


Figura 2.13: Red neuronal con arquitectura FeedForward densa. Las capas ocultas van desde $a^{(1)}$ a $a^{(L-1)}$ Fuente: Elaboración propia.

La arquitectura de una red neuronal corresponde al patrón de conectividad de las neuronas que la componen. Dependiendo de las restricciones aplicadas a la arquitectura se consiguen distintas implementaciones que se ajustan a las necesidades de un dominio en particular.

A continuación se describen algunas arquitecturas:

- Una **red Feed Forward** implementa una arquitectura que organiza las neuronas en capas tal que las neuronas de una misma capa no se comunican entre sí, sino que se conectan solo con neuronas de la capa anterior y la siguiente. Esta arquitectura implementa las siguientes capas: una o más capas de entrada que recibe los atributos que describen un ejemplo de entrada, capas intermedias (llamadas capas ocultas) y una o más capas de salida que entrega los valores que se buscan predecir. Tradicionalmente cada neurona de una capa oculta o de salida esta conectada a todas las neuronas de la capa anterior, lo que introduce una conectividad densa (Fully Connected).

Esta arquitectura permite la vectorización de las capas de la siguiente forma: el cálculo resultante de una capa (l) (llamado activación $a^{(l)}$) se representa con un vector de largo igual al número de neuronas de esa capa. Esta activación se obtiene al aplicar una función σ al resultado de multiplicar la activación de la capa $a^{(l-1)}$ (capa anterior) por una matriz $W^{(l)}$ y restando un vector $b^{(l)}$, tal como se describe en la ecuación (2.10). Ya que cada elemento de la capa corresponde a una neurona, la matriz W es una matriz de pesos entrenables tal de W_{ij} es el peso de conexión entre la neurona j de la capa

$(l - 1)$ y la neurona i de la capa (l) , mientras que el vector b es el vector de umbrales de activación. Lo usual es que σ sea una función no-lineal idéntica para todas las neuronas de esa capa. La capa de entrada no aplica una función ni contiene parámetros entrenables ya que sus unidades corresponden a los atributos que describen un ejemplo X . En la Figura 2.13 se muestra un ejemplo de red FeedForward.

$$a^{(l)} = \sigma_l(W^{(l)}a^{(l-1)} - b^{(l)}) \quad (2.10)$$

- Una **Red Neuronal Convolucional (CNN)** implementa una arquitectura "*diseñada para trabajar con inputs estructuras en cuadrícula (grid), que tienen fuertes dependencias espaciales en las regiones locales de la cuadrícula*" [Aggarwal, 2018]. Esta arquitectura es muy popular en contextos donde hay alta dimensionalidad y los datos vecinos están altamente correlacionados formando patrones locales. Ejemplos de estos contextos son los problemas de visión computacional, procesamiento de lenguaje natural y series de tiempo. Para lograr esto, estas redes siguen los siguientes principios de diseño:

- **Conectividad local:** a diferencia de las conexiones densas, esta conectividad restringe la vecindad de cada neurona a un subconjunto de los neuronas adyacentes a su entrada, lo que se conoce como **campo receptivo**. En el caso 1D, el tamaño del campo de una neurona son K posiciones del patrón de entrada, pudiendo desplazar los campos receptivos de neuronas adyacentes de acuerdo a un parámetro S llamado stride, donde la i -ésima neurona de la capa convolucional esta conectada con las posiciones $[S \cdot i, S \cdot i + K)$ del patrón de entrada. El número de neuronas de la capa convolucional depende del tamaño del input y los valores de K y S . En la Figura 2.14 se muestran dos ejemplos de capas convolucionales 1D con distintos valores de K y S , donde el número de neuronas varía para un mismo largo de entrada.

Es posible implementar un *pooling* o relleno que intenta mantener el tamaño de entrada en la capa convolucional. Para ello, en el caso 1D se rellena con ceros a cada lado de la entrada, donde el número de rellenos esta dado por las ecuaciones (2.11) y (2.12). Notar que si $S > 1$, el resultado puede no mantener el número de neuronas.

$$output_size = int(\frac{input_size + S - 1}{S}) \quad (2.11)$$

$$padding_needed = (output_size - 1) * S + K - input_size \quad (2.12)$$

Para el caso 2D, las neuronas de la capa convolucional se organizan en arreglos bi-dimensionales considerando dos valores de stride y dos valores del tamaño del campo receptivo (uno para cada eje).

- **Pesos compartidos:** consiste en que todos los campos receptivos compartan los mismos parámetros (pesos), lo que permite que un grupo de neuronas trabaje sobre un determinado patrón de entrada.

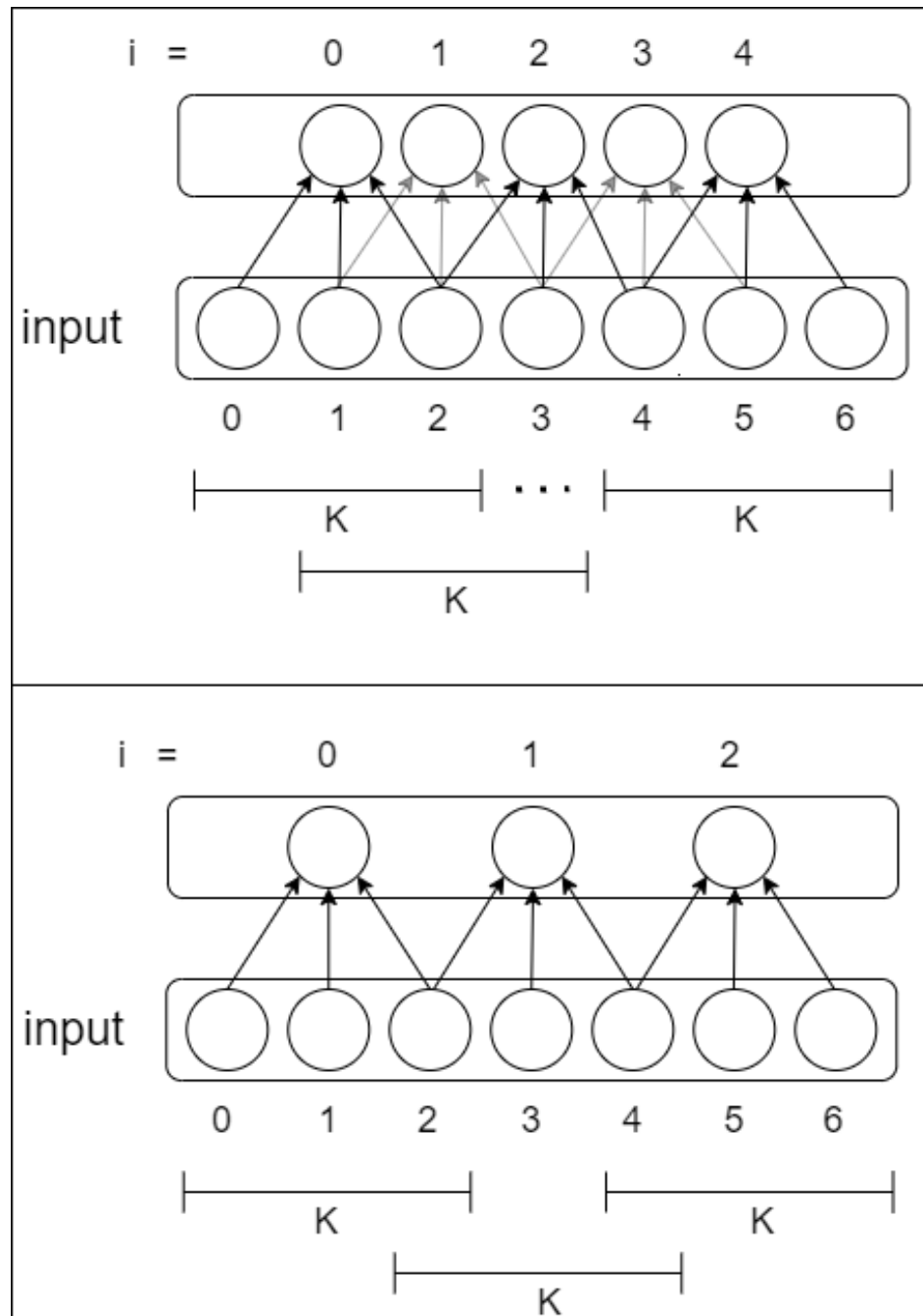


Figura 2.14: (superior) Ejemplo de capa convolucional 1D con $K = 3, S = 1$. (inferior) Capa convolucional 1D con $K = 3, S = 2$. En ambos el input es de largo 7. Fuente: Elaboración propia.

Esta característica se puede extender implementando múltiples grupos en una misma capa convolucional, donde cada grupo (llamados filtros o canales) comparten sus propios parámetros entrenables. En el caso 1D, el número de parámetros de la capa depende de K , el número de filtros y el número de filtros de

la entrada según la ecuación (2.13). Notar que el +1 corresponde al *bias*. Por último, la activación de un filtro recibe el nombre de *feature map*. Un ejemplo se presenta en la Figura 2.15.

$$n_parameters = n_filters * (n_filters_input * K + 1) \quad (2.13)$$

- **Pooling:** ante el posible aumento de dimensionalidad debido al uso de capas convolucionales se introduce una capa de Pooling, cuyo objetivo es reducir la dimensionalidad de su entrada. La reducción se logra al aplicar una estadística sobre campos receptivos (definidos por un tamaño y stride). Es común calcular el output como el valor máximo (Max Pooling) del campo receptivo asociado, manteniendo constante el número de filtros de la capa anterior. Este tipo de capa no posee parámetros entrenables.

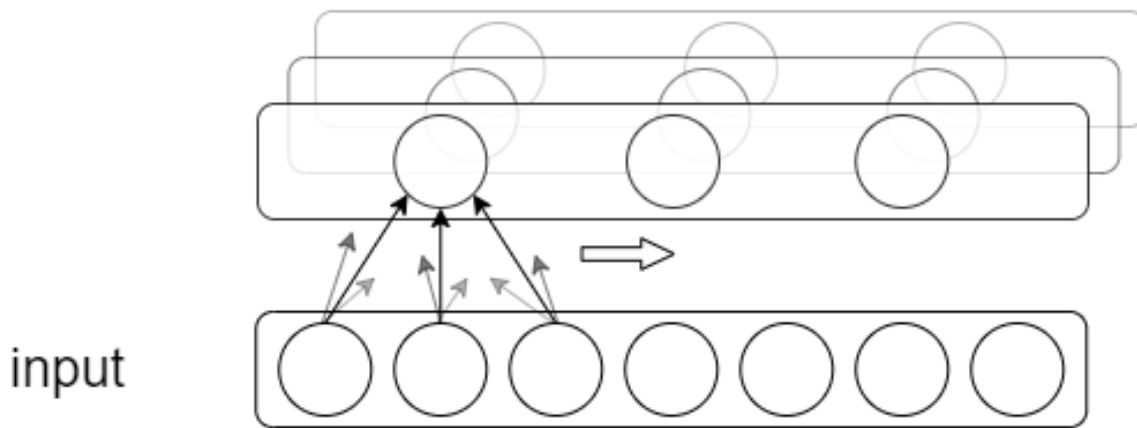


Figura 2.15: Ejemplo de capa convolucional ($K = 3$) con pesos compartidos y 3 filtros. El número de parámetros entrenables es 12 considerando los 3 bias asociados. Fuente: Elaboración propia.

En la práctica se pueden combinar capas densas con capas convolucionales de forma que el *feature map* de la última capa convolucional contenga las características más relevantes para la tarea y sean usadas como entrada de una arquitectura Feed-Forward densa. La unión entre una capa convolucional y una densa se realiza con una capa intermedia llamada **Flatten**, la cual toma los filtros de entrada y los reorganiza en un output unidimensional.

2.3.3. ENTRENAMIENTO

En un problema de clasificación lo habitual es que el número de neuronas de la capa de salida sea igual al número de categorías o clases del problema (o 1 menos), y que la predicción de cada neurona sea un valor numérico asociado a la probabilidad de que el ejemplo de

entrada pertenezca a dicha clase, es decir, que la salida de la red sea una distribución de probabilidad sobre las clases. Esto se puede lograr con una capa cuya función de activación es *softmax*, la cual se encuentra descrita en la ecuación (2.14), donde κ es el número de clases, ξ_k la preactivación de cada neurona de la capa de salida y o_i el vector de salida con las probabilidades predichas por la red del ejemplo i . Si se busca una distribución válida y el número de clases es 2, basta utilizar una sola neurona Sigmoidal de salida.

$$o_i = \frac{\exp(\xi_i)}{\sum_{k=1}^{\kappa} \exp(\xi_k)} \quad (2.14)$$

La tarea de entrenar una red neuronal consiste en determinar los valores de sus parámetros a partir de ejemplos. Este procedimiento comienza con la codificación de las etiquetas de cada ejemplo con la finalidad de hacerlas compatibles en una predicción. El método estándar es realizar un one-hot-encoding de las etiquetas, lo que consiste en transformar cada etiqueta en un vector de largo igual al número de clases, asignando un valor 1 en la j -ésima posición si el ejemplo pertenece a la j -ésima clase y asignando un valor 0 en el resto de posiciones.

Una vez codificadas las etiquetas, el entrenamiento requiere definir una función de costo L (*loss*) que permita cuantificar el error cometido por la red en su predicción. La función de costo *Categorical Cross-Entropy* es la más utilizada en problemas de clasificación con clases mutuamente excluyentes. Esta función está descrita en la ecuación (2.15), donde $[y_1, y_2, \dots, y_{\kappa}] \in \{0, 1\}$ es el vector one-hot correspondiente a la etiqueta de un ejemplo i y o_i el vector de salida de la red con las predicciones sobre i . Si el número de clases es 2, la función de costo es llamada Binary Cross Entropy.

$$L = - \sum_{k=1}^{\kappa} y_k \log(o_k) \quad (2.15)$$

Con una función de costo definida, se entrena la red con el objetivo de minimizar esta función mediante la modificación de los parámetros entrenables en dirección del negativo del gradiente de L respecto a cada parámetro. Este movimiento permite un ajuste de un parámetro proporcional a su influencia sobre la función objetivo. Esto se logra mediante iteraciones (llamadas *epochs*, diferente al concepto de *epoch* de *phase-folding*) sobre el conjunto de ejemplos de entrenamiento según la ecuación (2.16), donde $t = 1, 2, 3 \dots$ corresponden a los *training epochs* y θ son los parámetros entrenables de la red. η_t es una constante que controla el movimiento (tasa de aprendizaje). Si N es el número de ejemplos a utilizar, el gradiente se puede calcular respecto a cada ejemplo de entrenamiento y luego promediando los resultados tal como se describe en la ecuación (2.17), donde $(x^{(i)}, y^{(i)})$ es el ejemplo de entrenamiento i , $f(x^{(i)})$ es la predicción de la red y $\theta_j^{(l)}$ el parámetro j de la capa (l) . Esta expresión es resultado de considerar el objetivo como minimizar el valor esperado (aproximado) de la *loss* con un conjunto finito de ejemplos de entrenamiento tal como se describe en la ecuación (2.18). Este objetivo es llamado minimizar el error de entrenamiento.

$$\theta^{(t+1)} = \theta^{(t)} - \eta_t \nabla L(\theta^{(t)}) \quad (2.16)$$

$$\frac{\partial L}{\partial \theta_j^{(l)}} = \frac{1}{N} \sum_{i=1}^N \frac{\partial L(f(x^{(i)}), y^{(i)})}{\partial \theta_j^{(l)}} \quad (2.17)$$

$$E[L(f(x), y)] \approx \frac{1}{N} \sum_{i=1}^N L(f(x^{(i)}), y^{(i)}) \quad (2.18)$$

En el caso una red de múltiples capas, este procedimiento debe considerar que L esta en función de los parámetros en múltiples capas anteriores, luego, *“el gradiente se calcula usando el algoritmo de Backpropagation que aprovecha la regla de la cadena del cálculo diferencial, la cual calcula los gradientes en términos de sumas de productos de gradientes locales en las diversas rutas desde un nodo (neurona) hasta la salida”* [Aggarwal, 2018]. En la práctica esto considera dos partes:

1. **Forward Pass:** ingresar un ejemplo de entrenamiento a la red, calcular el resultado de la capa (l) considerando los parámetros actuales (puede ser una solución inicial aleatoria), luego utilizar este resultado como input de la capa ($l + 1$). Este procedimiento empieza con la primera capa y termina una vez calculada la predicción de salida en la última capa.
2. **Backward Pass:** calcular el gradiente de L respecto a cada parámetro entrenable usando la regla de la cadena. Ya que los gradientes a calcular en la capa (l) necesitan los valores calculados en la capa ($l + 1$), se empieza desde la última capa y el algoritmo avanza hacia atrás aprovechando los cálculos realizados en capas superiores.

Estos pasos se pueden vectorizar con la finalidad de realizar los cálculos con un lote de ejemplos de entrenamiento de tamaño b llamado mini-epoch. Típicamente, se busca $b < N$, lo que permite ajustar los parámetros inmediatamente usando el gradiente promedio del mini-batch sin tener que esperar los gradientes de todos los ejemplos disponibles. Una vez todos los ejemplos de entrenamiento se utilizan (en sus respectivos mini-epoch) se da por completado un *epoch*. Si $b \ll N$ el procedimiento se denomina *Stochastic Backpropagation*, lo que realiza una aproximación del gradiente verdadero, donde si b es lo suficiente grande se cumple la ecuación (2.19).

$$\frac{\partial L}{\partial \theta_j^{(l)}} \approx \frac{1}{b} \sum_{i=1*}^b \frac{\partial L(f(x^{(i*)}), y^{(i*)})}{\partial \theta_j^{(l)}} \quad (2.19)$$

Una característica de una red neuronal es su alta capacidad para aproximar funciones, es mas, *"se ha demostrado que una red con una sola capa oculta de unidades no lineales (con un amplio rango elección de funciones como la unidad sigmoideal) y una única capa de salida (lineal) puede calcular casi cualquier función razonable"*. Como resultado, *las redes neuronales a menudo se denominan aproximadores de funciones universales.*" [Aggarwal, 2018]. Su expresividad permite gran cantidad de aplicaciones en diversos dominios, sin embargo, esto puede generar problemas si no hay especial cuidado en el entrenamiento.

Las redes neuronales son modelos que pueden presentar el problema **overfitting** o sobre-ajuste, es decir, pueden perder la capacidad de generalización sobre ejemplos que no fueron utilizados para entrenar debido a que el error de entrenamiento puede disminuir arbitrariamente al sobre-ajustar los parámetros según los datos de entrenamiento. La consecuencia directa de este problema es que el rendimiento de la red puede ser bajo al predecir con datos que no ha visto. En la tarea de detección de exoplanetas es de interés disminuir este error de predicción, por lo que a continuación se describen algunas metodologías y técnicas para prevenir este problema.

La metodología más sencilla es separar los datos disponibles en dos conjuntos: uno de entrenamiento (*train set*) y uno de pruebas (*test set*). El objetivo es entrenar múltiples modelos utilizando solo el *train set* y mantener independientes los datos del *test set* tal que estos datos no sean utilizados para calcular los parámetros del modelo. Al evaluar los modelos en el *test set* se obtiene un error de predicción más realista sobre datos futuros, sin embargo, si se modifican los modelos en base a su rendimiento en *test set*, se pierde independencia de los conjuntos y se vuelve a presentar **overfitting**. Para seleccionar modelos manteniendo la independencia del *test set* se separa el *train set* en dos conjuntos: un nuevo *train set* y uno de evaluación (*validation set*). El objetivo de utilizar los 3 conjuntos es entrenar modelos con el *train set*, elegir el mejor modelo en base a su rendimiento en el *validation set* y finalmente evaluar su rendimiento sobre el *test set*.

Un problema que puede surgir al usar esta metodología es la elección de un conjunto de validación no representativo o que al comparar dos modelos o más modelos, uno de ellos tenga mejor rendimiento sobre la validación pero un mal rendimiento sobre los datos finales. Entre las técnicas que enfrentan este problema se encuentra el **K-fold Cross Validation**. Esta técnica consiste en dividir los datos que no son del *test set* en K bloques de igual tamaño para despues entrenar los modelos usando como *train set* los ejemplos de $K - 1$ bloques, considerando como *validation set* el bloque restante. Iterativamente se van rotando los bloques tal que cada bloque sea *validation set* una vez. Al final de cada iteración se registra el rendimiento del modelo utilizando una métrica. Finalmente se elige el modelo con un mejor rendimiento promedio y se vuelve a entrenar con todos los datos disponibles que no son del *test set*.

2.3.4. MÉTRICAS, REGULARIZACIÓN Y TASAS ADAPTATIVAS

Sobre la **evaluación de modelos**, en la práctica se utilizan distintas métricas aparte del *loss* de entrenamiento o de predicción con fines de interpretabilidad, decisión y aplicación en el contexto de la tarea. A continuación se describen algunas métricas en la tarea de clasificación, donde TP , FP , FN y TN son el número de ejemplos verdaderos positivos, falsos positivos, falso negativo y verdaderos negativos, respectivamente.

- **Precisión:** fracción de predicciones correctas del modelo respecto a una clase c . En contexto, es *"la fracción de señales clasificadas como planetas que son verdaderos planetas"* [Shallue y Vanderburg, 2018].

$$P_c = \frac{TP}{TP + FP} \quad (2.20)$$

- **Recall:** fracción de ejemplos una clase c de interés que el modelo efectivamente reconoce. En contexto, es *"la fracción de planetas verdaderos que se clasifican como planetas"* [Shallue y Vanderburg, 2018].

$$R_c = \frac{TP}{TP + FN} \quad (2.21)$$

- **Especificidad:** fracción de ejemplos que no son de la clase c que el clasificador efectivamente reconoce. Se define la tasa de falsos positivos como $1 - S_c$.

$$S_c = \frac{TN}{TN + FP} \quad (2.22)$$

- **Accuracy:** fracción de predicciones correctas del modelo.

$$A = \frac{TN + TP}{TN + TP + FP + FN} \quad (2.23)$$

- **AUC-ROC:** Area Under The Curve (área bajo la curva) ROC (Receiver Operating Characteristics). Considerando distintos umbrales de decisión, se puede construir una curva de recall (R_c) v/s tasa de falsos positivos ($1 - S_c$), luego, el área bajo esa curva es la métrica AUC-ROC, también llamada AUC. En contexto, es *"la probabilidad de que un planeta seleccionado al azar obtenga una puntuación más alta que un falso positivo seleccionado al azar"* [Shallue y Vanderburg, 2018].

Sobre la **regularización**, las técnicas utilizadas para atenuar o prevenir el overfitting se denominan métodos de regularización. Muchos de estos métodos buscan restringir la complejidad efectiva del espacio de hipótesis que explora el modelo, es decir, limitar el tamaño del espacio de soluciones a explorar. En este escenario se entrena el modelo con una propiedad a-priori deseable.

- La técnica de regularización **Early stopping** consiste en detener el entrenamiento en cuanto se detecta el comportamiento de *overfitting*, usualmente monitoreando el rendimiento en el *validation set* al final de un *epoch*.

Sobre las **tasas adaptativas**, en redes neuronales las funciones objetivo son altamente no-convexas y la suavidad del gradiente puede cambiar a medida que nos movemos en el espacio de búsqueda. Es por esto por lo que se busca calibrar la tasa de aprendizaje n_t en función de que tan suave es el objetivo. Entre los métodos modernos para calibrar la tasa se encuentran Adagrad, Adadelta, RMS-Prop y Adam, los cuales se adaptan a la curvatura de la función objetivo evitando divergencia en regiones muy curvas y acelerando el paso en regiones más planas. Un pseudocódigo con sus funcionamientos se encuentra en el Anexo 6.1. ALGORITMOS DE TASAS ADAPTATIVAS. A continuación se describen:

- **Adaptative Gradient o Adagrad:** Este método es base para el resto de algoritmos descritos e incorpora una tasa diferente por cada parámetro. Para cada parámetro mantenemos información de las magnitudes de los gradientes utilizados en un parámetro r que cambia en el tiempo.
- **Root Mean Squared Error Propagation o RMS Prop:** este algoritmo sustituye la suma que calcula Adagrad por una media móvil que permite olvidar los movimientos pasados según el valor del hiper-parámetro ρ (rho). Un valor de ρ cercano a 1 hace que el método olvide suavemente los gradientes pasados, es decir, se concentra en la localidad.
- **Adaptative Moment Estimation o Adam:** esencialmente es RMS Prop combinado con momentum. La velocidad v y r se calculan con media móvil. Existe una modificación de este algoritmo que enfrenta un problema durante las primeras iteraciones, en donde la estimación del gradiente es sesgada.

2.3.5. APLICACIONES

El estudio de los datos de la misión BRITE y K2 fue explorado por [D. Andrešič, 2021] con la finalidad de utilizar redes neuronales artificiales en la tarea de clasificación de estrellas variables (en 6 clases: *Delta Scuti—young*, *Detached Eclipsing Binary*, *Semi-Detached/Contact Eclipsing Binary*, *Gamma Dor—young*, *RR Lyrae ab—pulsating* y otras periodicidades). Para ello realizan un pre-procesamiento de los datos donde balancea los datos para tener la misma cantidad de estrellas variables y no-variables, luego corta las curvas de luz para que tengan la misma longitud, mezcla los datos, generan periodogramas, realizan transformaciones de Fourier, reduce la dimensionalidad de los atributos mediante la extracción de atributos significativos, realiza una normalización de datos en intervalos relevantes para la función de activación seleccionada y separa los datos en *train* y *test set*. Finalmente analiza los resultados al clasificar con MLP y LSTM con distintas funciones de activación, no superando el 65 % de *accuracy*. Concluyen que los malos resultados probablemente se deben al pequeño tamaño del conjunto de datos y los intervalos irregulares entre mediciones temporales. Cabe mencionar que a la fecha (2023) hay mayor cantidad de datos BRITE disponibles.

El problema de búsqueda de exoplanetas con redes neuronales sobre los datos de la misión BRITE es estudiado por [Yeh y Jiang, 2020] inyectando señales sintéticas sobre los datos originales para producir curvas de tránsitos según el modelo Mandel-Angol y detectarlas con modelos 1D-CNN. A continuación se resume el procedimiento realizado:

Los autores trabajan con el *Data Release* (DR) 2 y 3 del proyecto BRITE, considerando que las observaciones a una misma estrella hechas por distintos nanosatélites corresponden a curvas de luz separadas, mientras que distintas curvas de una estrella producidas por el mismo nanosatélite son combinadas en una sola curva. Para reducir la dispersión en las curvas, evitan realizar las correcciones propuestas en [Popowicz *et al.*, 2017] por motivos de eficiencia, proponiendo el procedimiento descrito a continuación, resultando en 35 curvas de luz (una por estrella):

1. Dividir los puntos de la curva en intervalos de 20 minutos (*bins*).
2. Calcular el flux promedio de cada intervalo.
3. Normalizar los datos anteriormente agrupados con respecto a los promedios de las observaciones.
4. Eliminar *outliers* considerando 3 desviaciones estándar.

Los parámetros incluidos para la inyección de tránsito son periodo P , el ratio entre el semieje mayor orbital y el radio estelar (a), el ratio entre el radio del planeta y el radio estelar (R_{ps}), inclinación orbital (i) y los coeficientes de *limb darkening* (μ_1, μ_2). Estos valores se eligen aleatoriamente siguiendo una distribución uniforme en los intervalos presentes en la Tabla 2.5 a excepción del periodo, cuyo valor depende del modelo de detección implementado.

Parameter	Values
a	10-30
R_{ps}	0.06 - 0.15
i	86 - 90°
μ_1	0.5
μ_2	0.0

Tabla 2.5: Rango de valores usados para el algoritmo de Mandel y Agol. Fuente: [Yeh y Jiang, 2020]

Utilizan el método *phase-folded* para generar un gran número de curvas variando el periodo (*trial period*) argumentando que la mayoría de posibles señales de tránsitos presentes en los datos originales es destruida. Esto se fundamenta según lo detallado en el Capítulo 2.1.3 para curvas *phase-folded* con un periodo incorrecto. Para permitir a los modelos detectar tránsitos en curvas con un periodo ligeramente diferentes al periodo verdadero del tránsito, crean dos curvas *phase-fold* extra con periodos dos minutos más cortos y dos minutos más largo que P , esto es $P \pm 2/1440[days]$. La mitad de los datos generados corresponden a curvas con tránsito y la otra mitad curvas sin tránsito, de los cuales 80 % se reserva para el *train set*, 10 % para *validation set* y 10 % para *test set*.

Respecto a los modelos, implementan 4 modelos con arquitectura convolucional (1D-CNN). El primero de ellos con la tarea de detección de tránsitos con periodo de 1 a 2 días, el siguiente modelo para periodos de 2 a 3 días, y así sucesivamente hasta el último modelo para los periodos entre 4 y 5 días. De esta forma, los valores de periodo usados en el algoritmo Mandel y Agol se distribuyen uniformemente según el rango de cada modelo. Cada uno de estos modelos es entrenado 5 veces. En la Figura 2.16 se presenta la arquitectura CNN usada para periodos de 1 a 2 días.

Los autores utilizan las métricas *accuracy*, *reliability* (precisión) y *completeness* (*recall*) para evaluar los resultados, obteniendo valores mayores a 99.76 %, 99.78 % y 99.63 %, respectivamente para todos los modelos. Finalmente obtienen 10 candidatos a exoplanetas, de los cuales dos (sistemas HD37468 y HD186882) tienen una mayor prioridad para una futura investigación.

La búsqueda de exoplanetas con redes neuronales sobre los datos de la misión TESS es explorada por [Yu et al., 2019], donde proponen el primer modelo CNN entrenado y testado con datos reales de TESS utilizando como base el *pipeline* y modelo propuesto por [Shallue y Vanderburg, 2018] (modelo AstroNet), en cuyo trabajo original identifican 2 nuevos exoplanetas en los datos de Kepler. A continuación se resume el procedimiento realizado por [Yu et al., 2019]:

Esta investigación busca realizar dos tareas relacionadas a la búsqueda de exoplanetas: (i)

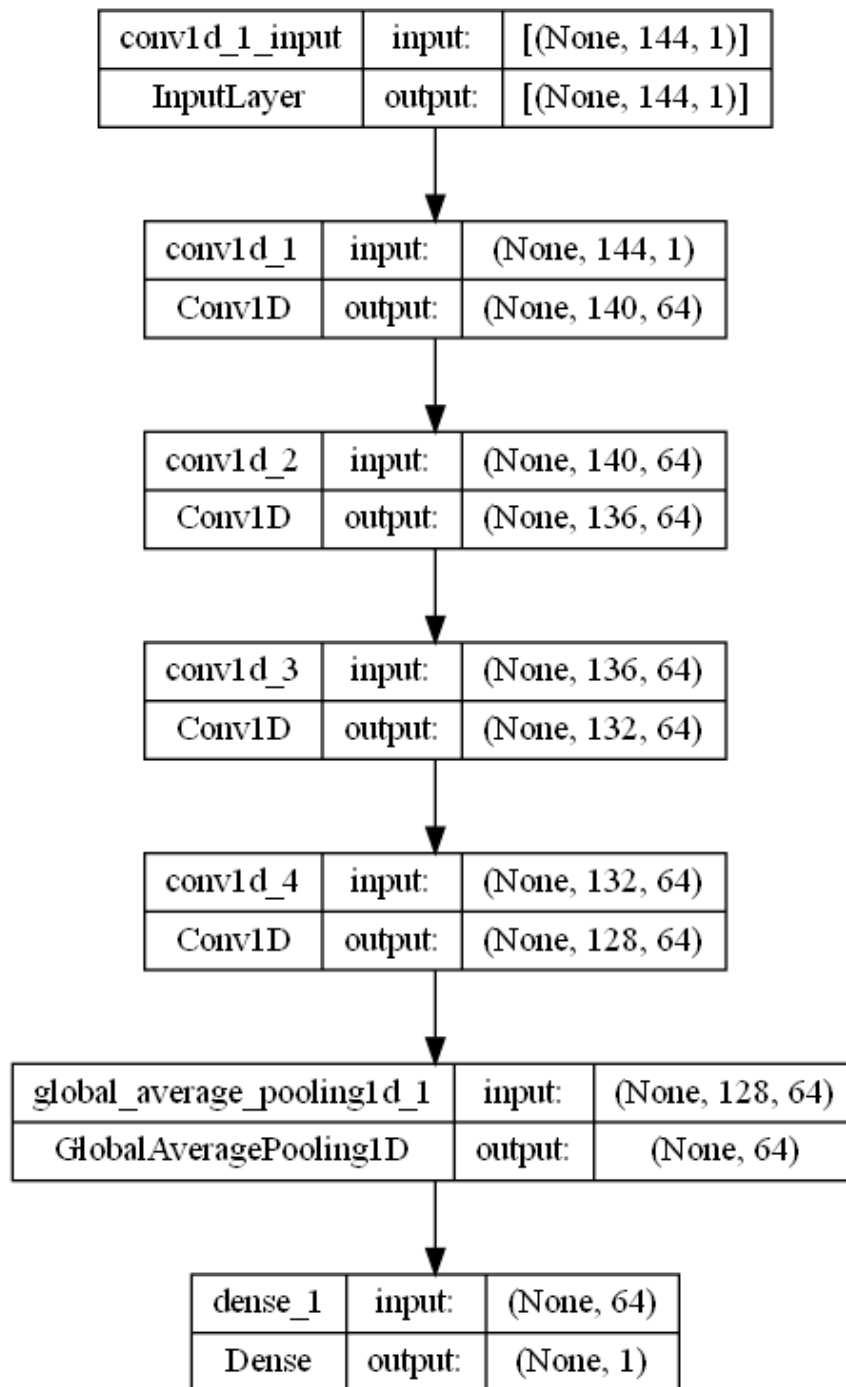


Figura 2.16: Arquitectura implementada para la tarea de detección de tránsitos con periodo de 1 a 2 días. Fuente: [Yeh y Jiang, 2020]

crear un triaje automático para eliminar todas las señales no planetarias obvias entre los TCE, y (ii) crear un *vetting* automático para diferenciar las señales encontradas en (i) en exoplanetas o binarias eclipsantes.

Para la tarea (i) consideran datos TCE etiquetados en 4 categorías: planeta candidato (PC), binaria eclipsante (EB), variabilidad estelar (V) y ruido instrumental (IS). Estas etiquetas son asignadas bajo criterio experto siguiendo 9 reglas de clasificación. En total trabajan con 16.516 TCEs para el triaje, lo que incluye 493 PCs, 2155 EBs, y 13,868 V y IS combinados. Tras aplicar un *random shuffle* sobre los datos, los separan en los conjuntos de entrenamiento (80 %), validación (10 %) y *testing* (10 %).

Los TCE utilizados son producto del MIT QLP, *pipeline* utilizado para la producción de curvas de luz y búsqueda de tránsito a partir en el TESS Input Catalog para objetivos en la banda TESS de magnitud más brillante que magnitud 13.5.

La representación de las curvas es realizada mediante *phase-folding* utilizando el periodo del TCE identificado previamente, con los tránsitos centrados en las curvas. Luego estas curvas se transforman en representaciones ***global view*** y ***local view***, cuyo procedimiento se detalla en el Capítulo 2.1.3. En la Figura 2.6 se grafica una curva usada en este trabajo, donde el *global view* considera un vector 1D de largo 201 y la *local view* un largo de 61. En la tarea de triaje cada ejemplo de entrenamiento consiste en este par de vectores, los cuales se utilizan como entrada para el modelo AstroNet, una arquitectura de red neuronal convolucional (1D-CNN) implementada con Tensorflow. Este modelo entrega una predicción entre 0 y 1, donde un valor cercano a 1 indica una alta confianza de que el ejemplo corresponde a un planeta. Esta arquitectura se presenta en la Figura 2.17.

Entrenan 10 copias idénticas del modelo AstroNet con diferentes parámetros de inicialización para después promediar sus predicciones, lo que se conoce como "*model averaging*". Utilizando un umbral de decisión de 0.5 obtienen 97.4 % de Accuracy, 99.2 % AUC y 97.0 % Average Precision sobre el conjunto de *testing* para la tarea de triaje.

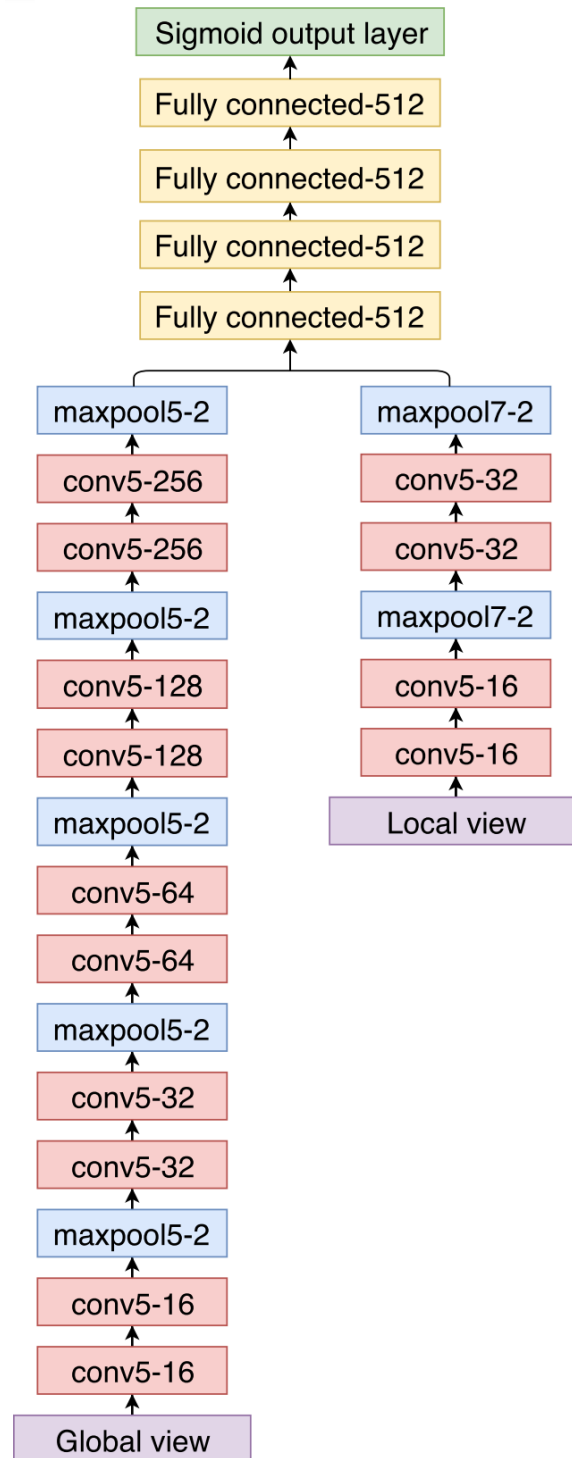


Figura 2.17: Arquitectura implementada en la tarea de triaje. Fuente: [Yu *et al.*, 2019]

CAPÍTULO 3

PROPUESTA DE SOLUCIÓN

3.1. PREPARACIÓN

3.1.1. LIBRERÍAS DISPONIBLES

Algunos de los métodos y algoritmos descritos en el Capítulo 3 se encuentran disponibles para su uso en el lenguaje de programación Python. En el contexto astronómico, la librería **astropy**⁷, ofrece funcionalidades claves y comunes para astronomía y astrofísica. Esta incluye la clase **LombScargle** para computar periodogramas Lomb-Scargle y constantes útiles como el radio del Sol y radios de los planetas Tierra y Júpiter.

El paquete **PyTransit**⁸, usada para el modelamiento optimizado de tránsitos, posee una implementación modelo cuadrático de Mandel y Agol mediante la clase **QuadraticMode**.

PyAstronomy⁹ es una colección de paquetes relacionados a astronomía. Es de interés en este trabajo su proyecto **Python AstroLib (pyasl)**.

Wotan es un paquete open-source de Python que ofrece un toolkit para detrending de series de tiempo en el contexto de tránsitos exoplanetarios [Hippke *et al.*, 2019].

En el contexto de Deep Learning, **TensorFlow**¹⁰ es una plataforma de código abierto para el aprendizaje automático. Facilita la creación de modelos de aprendizaje automático para computadoras de escritorio, dispositivos móviles, la web y la nube.

Keras¹¹ es una API de Deep Learning escrita en Python, que se ejecuta sobre la plataforma de TensorFlow. Permite crear prototipos rápidamente, llevar a cabo investigaciones y producir resultados con una API fácil de usar para definir y entrenar Redes Neuronales Artificiales.

En el contexto de ciencia de datos, las siguientes librerías son recurrentes: **NumPy**¹² es una librería fundamental para la computación científica en Python. **Pandas**¹³ es una librería que proporciona estructuras de datos y herramientas de análisis de datos de alto rendimiento y fáciles de usar en Python.

⁷<https://docs.astropy.org/en/stable/index.html>

⁸<https://pytransit.readthedocs.io/en/latest/>

⁹<https://pyastronomy.readthedocs.io/en/latest/index.html>

¹⁰<https://www.tensorflow.org/>

¹¹<https://keras.io/about/>

¹²<https://numpy.org/>

¹³<https://pandas.pydata.org/>

3.1.2. DATOS DISPONIBLES

La tarea de detección de posibles exoplanetas se realiza sobre los datos públicos del proyecto BRIT¹⁴ utilizando el lenguaje de programación Python. Las curvas de luz públicas se encuentran disponibles para su descarga en la página Wiki del proyecto¹⁵, sitio que aloja toda la información de interés para trabajar sobre los datos de BRIT.

En la Wiki se organizan los datos públicos en un árbol de archivos con dos carpetas principales: las curvas públicas y los archivos fit con las imágenes full frame. Las curvas de luz se organizan por sectores (64 sectores a la fecha 26/07/2022), los cuales tienen un identificador único. Por ejemplo, "22-Ori-IV-2016_DR5" corresponde a la cuarta observación en el sector Orión dentro del *Data Release 5*. Al no existir una funcionalidad para descargar automáticamente todas las curvas de luz, se programa un *script* para (i) buscar todos los enlaces que alojan los archivos, y (ii) descargarlos automáticamente.

Se utilizan las curvas de los *Data Release*: DR2, DR3, DR4, DR5 y DR6, donde hay datos disponibles para 609 estrellas diferentes en 4063 archivos. Cada archivo es un texto plano (ASCII) con una extensión .dat. Los archivos contienen dos secciones: el *header* y los datos. La sección *header* contiene metadatos asociados a la observación (información general del nanosatélite, instrumento, configuración, etc.), seguido de la descripción de las columnas presentes en los datos. Los metadatos de mayor relevancia para esta investigación se encuentran en la Tabla 3.1. La sección de datos se presenta en formato tabla, donde los valores se encuentran separados por espacios en blanco. El número de columnas que proporciona cada archivo depende del *Data Release* que corresponda, tal como se muestra en la Tabla 3.2.

¹⁴"Based on data collected by the BRIT Constellation satellite mission, designed, built, launched, operated and supported by the Austrian Research Promotion Agency (FFG), the University of Vienna, the Technical University of Graz, the Canadian Space Agency (CSA), the University of Toronto Institute for Aerospace Studies (UTIAS), the Foundation for Polish Science Technology (FNP MNISW), and National Science Centre (NCN)."

¹⁵<http://brite-wiki.astro.uni.wroc.pl/bwiki/doku.php?id=start>

ID	Description	Example
SatellID	satellite short ID	BHr
SatLauDa	satellite orbiting period [min]	97.097
SatFilte	filter information	red
ObsMode	ch : chopping horizontally along X axis, cv : chopping verically	ch
FieldIDn	observation field ID	25-VelPup-III-2017
ReleaseV	release version	DR5
ROIxsiz	Raster X size [pixel]	52
ROIysiz	Raster Y size [pixel]	28
ROIxpos	Raster X position [pixel]	1309
ROIypos	Raster Y position [pixel]	2065

Tabla 3.1: Tabla con los valores de headers más relevantes para esta investigación. Fuente: elaboración propia.

En consecuencia a los problemas descritos en el subcapítulo 2.2.3 (MISIÓN BRITE), solo se utilizan los datos creados con el modo de observación “*chopping*”, es decir, aquellos con valor *ch* (*chopping horizontally*) o *cv* (*chopping verically*) en el campo *ObsMode* del *header*. El número de archivos que cumplen esta restricción es 3147, los cuales se agrupan bajo el nombre de **dataBRITE_chop**.

3.1.3. PREPROCESAMIENTO

El equipo científico de BRITE [Pigulski, 2018] recomienda realizar un preprocesamiento de los datos de BRITE antes de trabajar sobre las curvas de luz debido a tres factores principales:

- El número de *hot pixels* y otros defectos de chips son altos y crecen con el tiempo.
- El apuntamiento a veces no es ideal, lo que resulta en un bamboleo de las estrellas en los *rasters*.
- Un defecto adicional de CTI en algunas regiones de los detectores.

El *pipeline* de preprocesamiento sugerido no presenta un código para una aplicación directa, por lo que hay que interpretar las instrucciones y programar. A continuación se resume los pasos recomendados y detalles de implementación:

Column	Description	Data Release
HJD	Heliocentric Julian Date at start of exposure	2,3,4,5,6
FLUX	Signal extracted from image per second [ADU/s]. FLUXAPT for DR6.	2,3,4,5,6
CCDT	CCD Temperature [°C]	2,3,4,5,6
XCEN	Profile centre of gravity with respect to raster origin [pixel]	2,3,4,5,6
YCEN	Profile centre of gravity with respect to raster origin [pixel]	2,3,4,5,6
JD	Julian Date at start of exposure as listed in FITS header	2,3,4,5,6
PSFC1	PSF blurring coefficient 1	3,4,5,6
PSFC2	PSF blurring coefficient 2	4,5,6
RTSC	RTS column indicator	3,4,5,6
RTSP	RTS pixel indicator	5,6
APERO	Aperture Offset	5,6
APERF	Aperture Out of Raster indicator: 1=OK, 0=out-of-raste	5,6

Tabla 3.2: Tabla con las columnas presentes en los datos. Los datos en DR6 poseen más columnas, pero no se utilizan durante el procesamiento de datos en este trabajo. Fuente: elaboración propia.

1. **Reformateo** de los *raw fluxes* y el tiempo. Se utiliza la ecuación (3.1) para calcular la magnitud usando la columna FLUX o FLUXAPT (DR6), y se resta el valor 2456000.0 a la columna HJD (*Heliocentric Julian Date at start of exposure*) y se le suma medio tiempo de exposición. Esta operación facilita la visualización e interpretación en el eje temporal, ahora en días. En adelante este resultado se denomina tiempo o *Time*. En la Figura 3.1 se muestran dos curvas con este paso aplicado.

$$Magnitude = -2,5 \cdot \log_{10}(FLUX) + 14,4 \quad (3.1)$$

2. **Remover de los datos los outliers extremos** para cada columna ('Magnitude', 'XCEN', 'YCEN', 'CCDT', 'PSFC1', 'PSFC2', 'RTSC', 'RTSP', 'APERO'). Los autores proponen el uso de las técnicas *Generalized Extreme Studentized Deviate (GESD)* y *sigma clipping*. En esta implementación particular se utiliza *sigma clipping* con un factor de 3 para todas las columnas. En la Figura 3.2 se muestran algunos outliers detectados.

Debido a la naturaleza de las observaciones en modo “*chopping*”, las columnas ‘XCEN’ e ‘YCEN’ son casos especiales. En ambos casos los valores se agrupan en 2 *clusters*, tal como se muestra en la Figura 3.3. La remoción de outliers se realiza por cada *cluster* por separado. La frontera de separación entre ambos *clusters* se define en 0.5 ROIxsiz para el caso horizontal y 0.5 ROIysiz para el vertical. Ambos valores ROIxsiz y ROIysiz son proporcionados en el *header* y no son necesarios para los datos DR6 ya que la frontera de separación se encuentra en el valor cero.

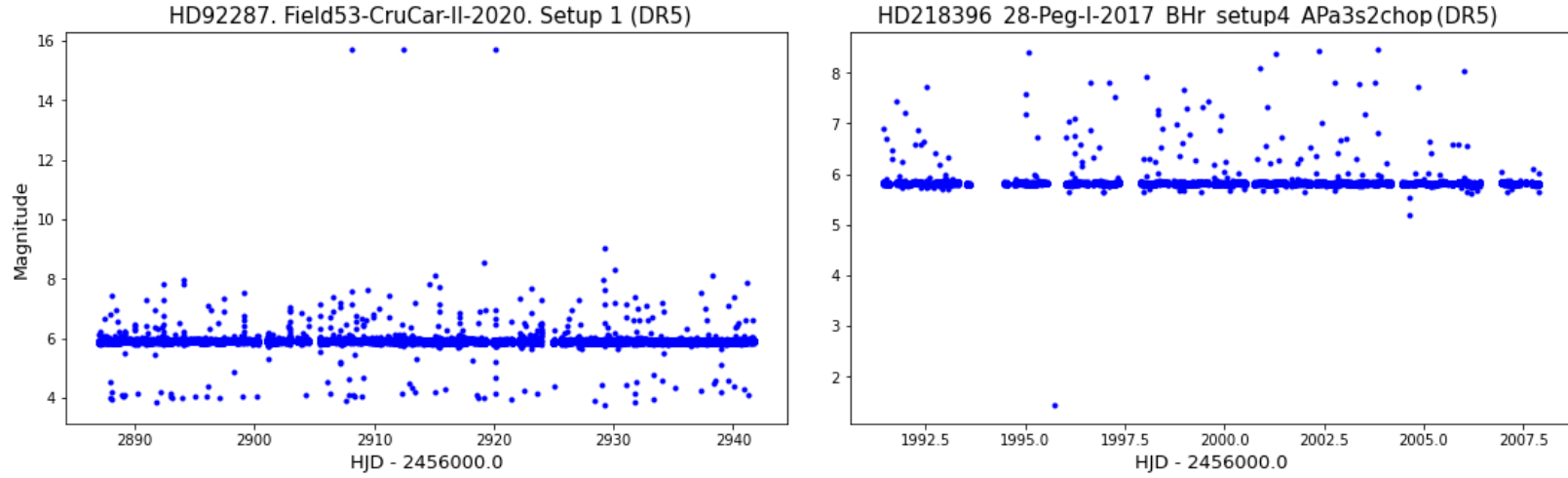


Figura 3.1: Dos curva de luz resultantes del reformato de los *raw fluxes* y el tiempo. Las estrellas corresponden a HD92287 y HD218396. Fuente: Elaboración propia.

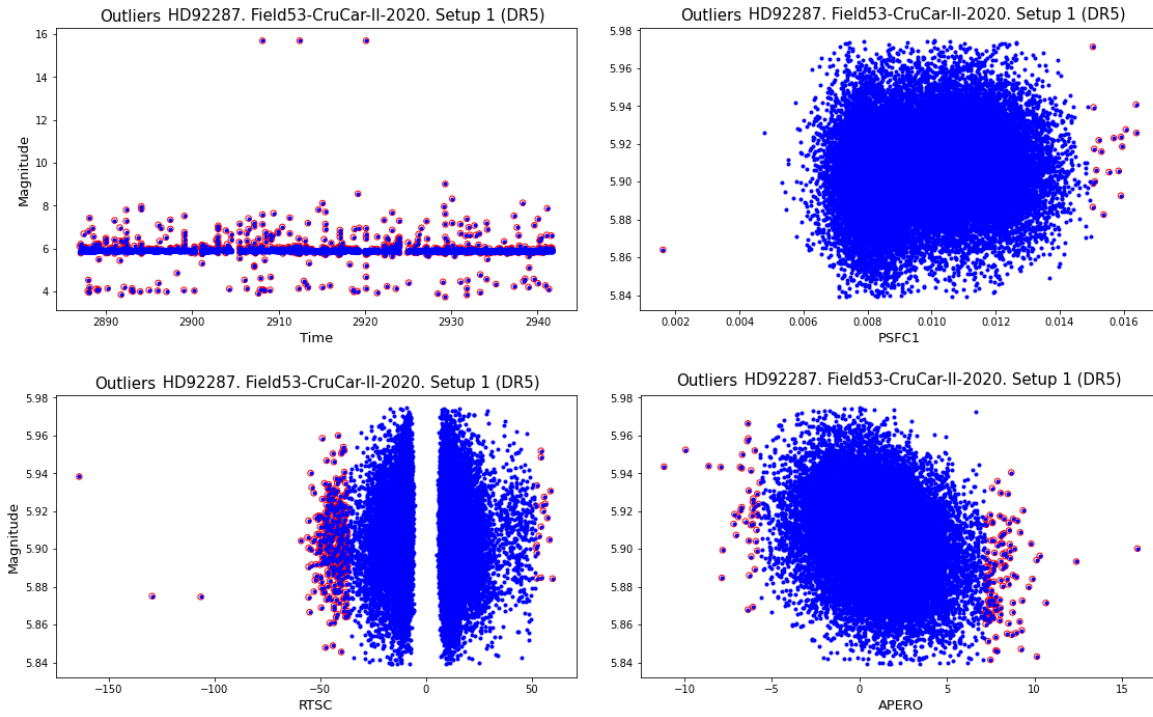


Figura 3.2: Ejemplos de outliers (marcados con círculos rojos) para las columnas 'Magnitude', 'PSFC1', 'RTSC' y 'APER0'. Se utiliza el *sigma clipping* con valor de $\alpha = 3$. Fuente: Elaboración propia.

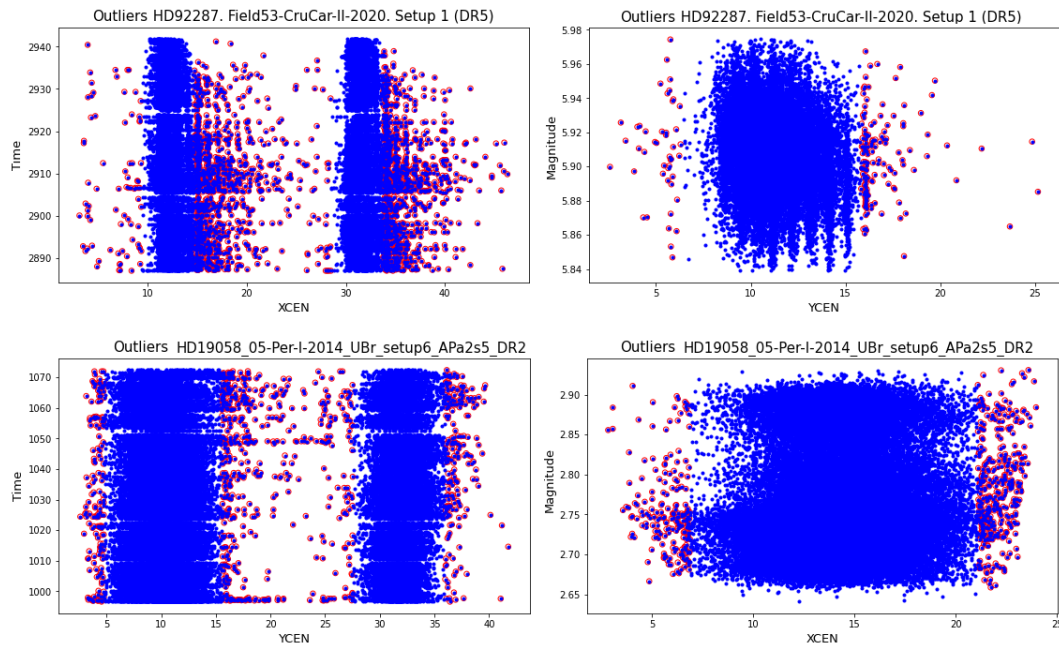


Figura 3.3: Ejemplos de *clusters* en la columna 'XCEN' e 'YCEN'. Los outliers son marcados con círculos rojos. Fuente: Elaboración propia.

3. **Remove outliers por órbita** para cada columna. Los datos originales no indican qué dato corresponde a qué órbita. Para obtener eso diseña una implementación propia *naive* para separar los datos en órbitas:

Método *naive*:

- Asumir órbita circular.
- Asumir que la toma de datos sólo puede realizarse en a lo más en media órbita (supuesto realista ya que la toma de datos es menor a la media órbita).

Este método permite separar órbitas de forma general sin conocer todos los detalles de la órbita real ni la sección de órbita utilizada durante la recolección de datos al solo requerir el periodo orbital, el cual es proporcionado en el *header* o por [Pablo *et al.*, 2016]. En las Figuras 3.4 y 3.5 se presentan diagramas que describen el método propuesto. Se puede asegurar que si la toma de datos empieza entre A y B, durante los próximos Δ minutos, no empezará la siguiente órbita. En consecuencia, los datos que pertenecen a la i -ésima ($i \in [0, last_orbit]$) órbita cumplen con la condición presente en la ecuación (3.2), donde $\max(Time)$ y $\min(Time)$ corresponden al menor y mayor tiempo presente en los datos. El índice de la última órbita ($last_orbit$) se calcula con la ecuación 3.3. El valor de Δ es medio periodo orbital ($orbital_period$).

$$\{data[Time] \geq (start + i * orbital_period)\} \& \{data[Time] \leq (start + (i + 1) * orbital_period)\} \quad (3.2)$$

$$last_orbit = int\left(\frac{\max(Time) - \min(Time) + \Delta}{orbital_period}\right) \quad (3.3)$$

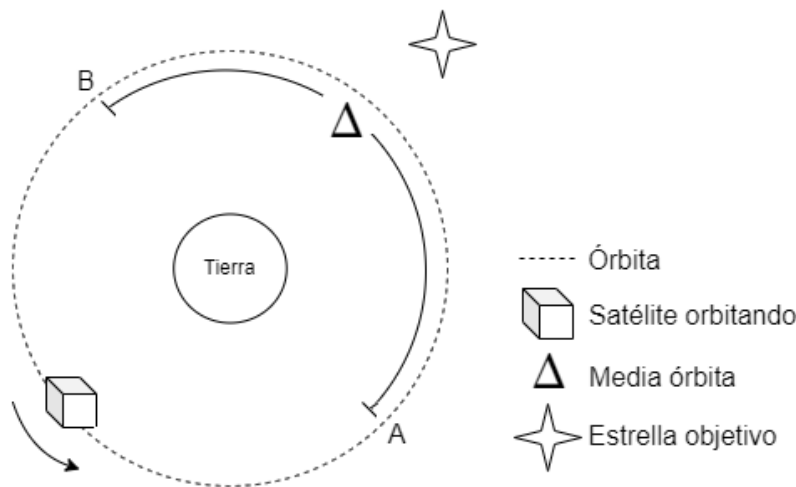


Figura 3.4: Diagrama orbital del método *naive* propuesto. Fuente: Elaboración propia.

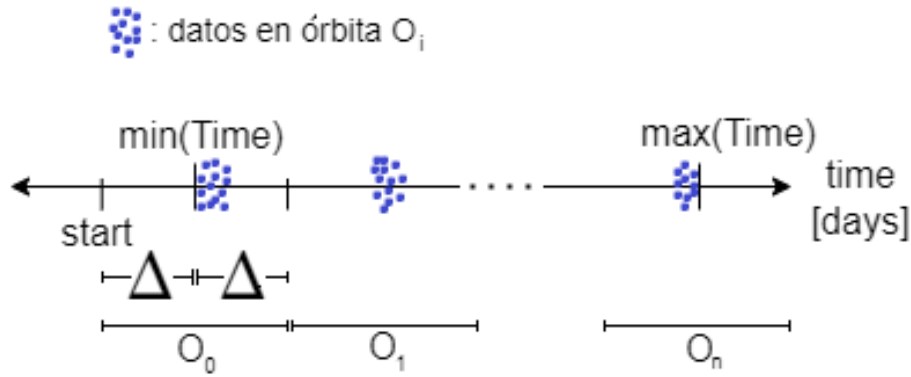


Figura 3.5: Ejemplo de distribución de datos por órbita en el método *naive* propuesto. Fuente: Elaboración propia.

4. **Remove peores orbitas**, es decir, aquellas con mayor dispersión medida por desviación estándar respecto a cualquier columna. Se utiliza el método *naive* anteriormente descrito para esta tarea.
5. Sustraer cualquier señal intrínseca fuerte. Este paso no se realiza.
6. **Decorrelación**: las magnitudes originales están correlacionadas con la temperatura de la cámara, la posición del centroide, fase orbital y otros parámetros. Los autores describen que la correlación es el efecto instrumental más significativo en la fotometría de BRITe y en consecuencia proponen un procedimiento 1D y 2D para la decorrelación, considerando que el primero tiene un mayor efecto que el segundo. La decorrelación 1D considerando un parámetro (columna) P posee los siguientes pasos:
 - a) considerar los datos magnitud en función del parámetro P .
 - b) ajustar una curva que describa la dependencia. Se sugiere la interpolación Akima.
 - c) corregir la correlación al restar la función ajustada. Es decisión del programador remover o no los *outliers* sobre el resultado.

Se aplican estos pasos para los parámetros 'CCDT', 'PSFC1', 'PSFC2', 'RTSC', 'RTSP', 'APE-RO' y 'Phase' (para DR2, DR3 y DR4 se aplica a un subconjunto de estos parámetros). El nuevo parámetro 'Phase' (fase) se incorpora ya que se espera ver efectos instrumentales relacionados con la fase orbital, sin embargo, este no es proporcionado en los datos originales y debe ser calculado por el programador. En esta implementación particular, la fase del dato j en la órbita i se calcula siguiendo la ecuación (3.4) tal que $Phase_j \in [0, 1]$.

$$Phase_j = \frac{Time_j - (start + i * orbital_period)}{orbital_period} \quad (3.4)$$

Se utiliza la implementación de interpolación Akima de la librería `scipy`¹⁶. Esta clase se ajusta (*fit*) con dos vectores correspondientes a los a puntos en el eje x y puntos en el eje y respectivamente ($\text{dimension}(x) = \text{dimension}(y) = 1$). Los puntos de ambos ejes son llamados puntos ancla o *anchor_points*. En esta implementación se utilizan 60 pares de puntos ancla equiespaciados en el eje x . En la práctica, los puntos x corresponden a valores del parámetro P a decorrelacionar y los puntos y la media aritmética de los valores de la columna 'Magnitude' que caen entre dos puntos anclas del eje x consecutivos.

Una vez ajustada la función Akima, se interpola utilizando como entrada todos los valores de del parámetro P , obteniendo un vector definido con el nombre de *y_akima*. Finalmente se corrige la correlación restando *y_akima* a los valores de la columna 'Magnitude'.

Este procedimiento se puede aplicar una o más veces por cada parámetro (iteración). Dependiendo del orden de aplicación sobre los parámetros se obtienen diferentes resultados. El criterio para el orden de aplicación es empezar con la correlación más fuerte y luego continuar con las más débiles. Para medir la fuerza utilizan el parámetro R que es calculado según la ecuación (3.5), donde V_{pre} y V_{post} son las varianzas del parámetro 'Magnitude' antes y después de aplicar el procedimiento. Mientras mayor sea el valor de R , más fuerte es la correlación.

$$R = 100 * (1 - \frac{V_{post}}{V_{pre}}) \quad (3.5)$$

Los autores sugieren terminar el procedimiento una vez el valor de R es menor a 0.05 para cada parámetro, pudiendo variar el valor según lo defina el programador. En esta implementación particular se limita la cantidad de iteraciones a un máximo de 20 para acotar el tiempo de ejecución. Tras pruebas experimentales se descubre que el procedimiento tiende a estancarse en un solo parámetro, es decir, un parámetro siempre posee el menor valor de R , por lo que se implementa una heurística para no volver a corregir la correlación del parámetro P si la mejora respecto a la última iteración anterior del parámetro es menor al 5 %.

Finalmente, después de cada corrección, se remueven los outliers de 'Magnitude' utilizando *sigma clipping* con un factor de 4 (menos agresivo) y se remueven las peores órbitas según el procedimiento anteriormente descrito. En el Listing 3.1 se presenta un pseudocódigo que resume el procedimiento descrito y en la Figura 3.6 las primeras cuatro iteraciones aplicadas a la curva HD218396_28-Peg-I 2017_BHr_setup4_APa3s2chop_DR5.

El *pipeline* descrito se aplica sobre los archivos **dataBRITE_chop**, obteniendo 3139 curvas de luz. Las curvas resultantes están centradas en magnitud cero. Del total, 8 archivos gatillan una excepción en el *pipeline*. Estos se encuentran en el Anexo y se decide no utilizarlos.

¹⁶<https://docs.scipy.org/doc/scipy/reference/generated/scipy.interpolate.Akima1DInterpolator.html>

```

1
2 def apply_decorrelation(data):
3     # data corresponde a la curva a decorrelacionar
4
5     threshold = 0.95 # definir un valor umbral
6
7     for iteration in range(20): # iterar 20 veces
8         tabu_list = [] # crear una lista vacía
9         prev_Rs = {} # crear un diccionario vacío.
10
11         Calcular el valor de R para todos los parámetros
12
13         P = parámetro con el menor valor de R
14         R_min = menor valor de R
15
16         if (clave P in prev_Rs and R_min/prev_Rs[P] > threshold):
17             # agregar el parámetro P a la lista tabu si la mejora
18             # respecto a la última iteración anterior del parámetro
19             # P es menor al 5%
20             tabu_list.append(R)
21
22         else: # parametro P no se encuentra en lista tabu o la
23             # mejora es mayor a 5%
24
25             # ajustar y evaluar función Akima utilizando 60 puntos
26             Akima1DInterpolator = función Akima ajustada con P
27             y_akima = evaluación de Akima1DInterpolator
28                 usando data['Magnitude']
29
30             # corregir la correlación
31             data['Magnitude'] = data['Magnitude'] - y_akima
32
33             Remove outliers de data['Magnitude'] usando sigma
34             clipping con un factor de 4.
35
36             prev_Rs[P] = R_min # almacenar en el diccionario la llave P
37                 # con valor R_min, es decir, el valor R
38                 # calculado para ese parámetro.
39
40             Remove peores orbitas
41
42     return data

```

Listing 3.1: Pseudocódigo resumen de la decorrelación.

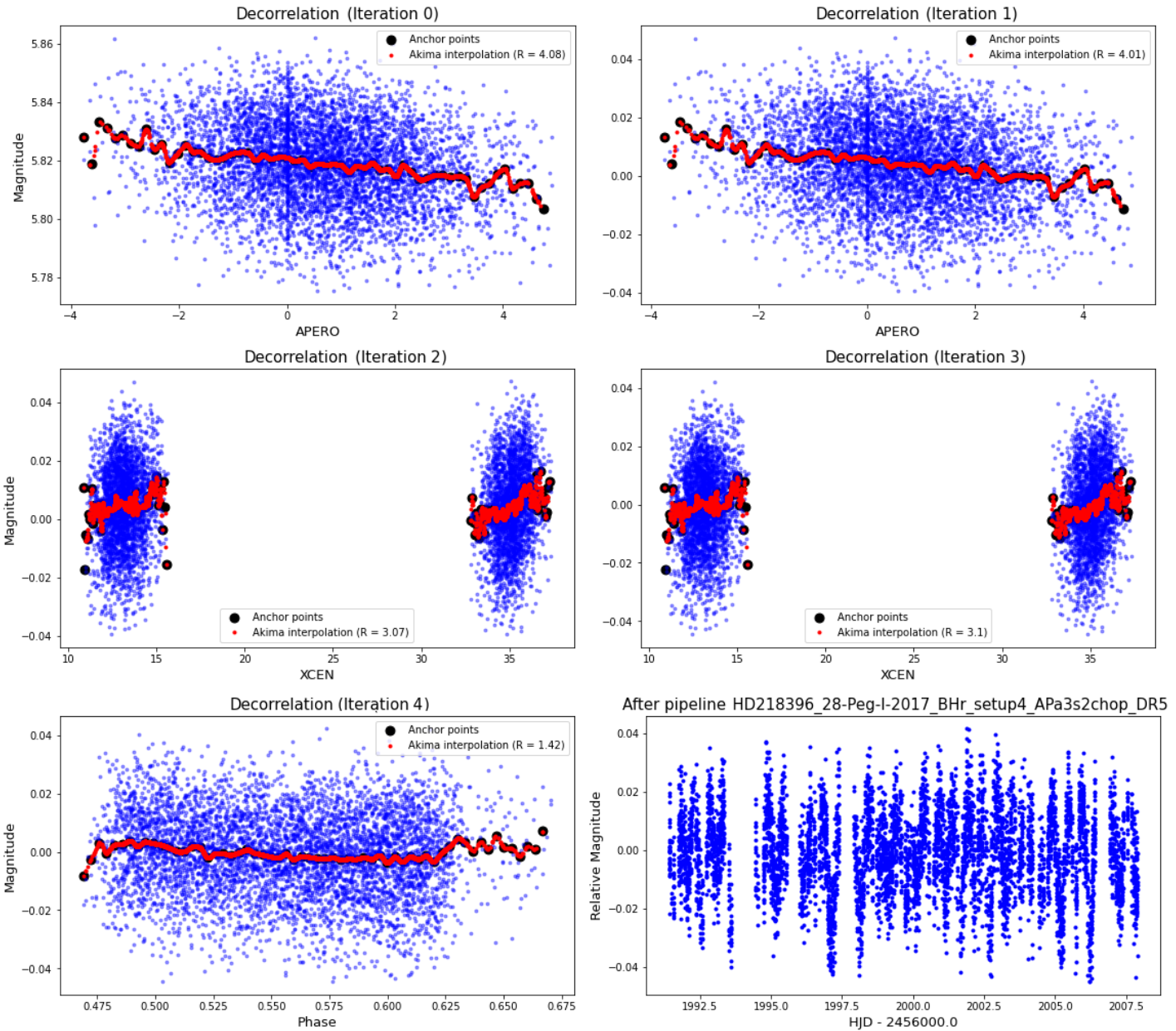


Figura 3.6: Primeras cuatro iteraciones de la implementación usada en la decorrelación. En las iteraciones 1 y 3 no se realiza la corrección. El resultado final tras 20 iteraciones se presenta en la esquina inferior derecha. Fuente: Elaboración propia.

3.2. CREAR DATASET

3.2.1. PREPARACIÓN

La misión BRITE no está diseñada para la búsqueda de exoplanetas, por lo que el proyecto original no proporciona datos similares a los eventos TCE¹⁷ que proporciona el proyecto TESS. En consecuencia, se adapta la estrategia de [Yeh y Jiang, 2020] para inyectar tránsitos sintéticos sobre los datos de BRITE descrito en el Capítulo 2.3.5. (APLICACIONES) con el objetivo de crear un nuevo dataset sintético de curvas con dos clases: **PC** (*Planet Candidate*) y **NOT_PC** (*Not Planet Candidate*) para la misión BRITE. Adicionalmente, para cada ejemplo se crean 3 representaciones: *phase fold*, *local view* y *global view*, utilizadas en el trabajo de [Yu et al., 2019] para la búsqueda de exoplanetas con redes neuronales sobre los datos de la misión TESS. Una misma curva BRITE concatenada puede ser usada para crear múltiples ejemplos al variar los parámetros de la inyección.

Se trabaja sobre las 3139 curvas de luz resultantes del *pipeline* descrito en el capítulo anterior. Las curvas son concatenadas si (i) corresponden a la misma estrella objetivo, y (ii) corresponden al mismo satélite. Este proceso resulta en 876 curvas de luz.

Se determina la proporción de ejemplos PC y NOT_PC tomando como base la proporción usada por [Yu et al., 2019], que considera 493 PCs (planeta candidato), 2155 EBs (binaria eclipsante) y 13,868 V (variabilidad estelar) y IS (ruido instrumental) combinados, con la diferencia que en este trabajo se utiliza como PC la suma de PC y EB, decisión fundamentada en (i) aumentar la cantidad de ejemplos PC y (ii) considerar que la tarea de detección es similar a la tarea de triaje de [Yu et al., 2019]. En consecuencia, 16,03 % de los ejemplos son PC y 83,96 % son NOT_PC.

Al combinar la inyección de tránsito con las 3 representaciones se presentan dos alternativas de aplicación:

1. Inyectar el tránsito directamente en las curvas preprocesadas y luego crear las 3 representaciones.
2. Preprocesar las curvas e inyectar el tránsito en la representación *phase fold*, luego crear las otras dos representaciones.

Se implementa la primera opción con el objetivo de preservar las ventanas de observación y su efecto en el tránsito, es decir, que la señal de tránsito solo se presente en tiempos donde efectivamente el satélite observa la estrella objetivo.

Un dato importante para la creación del dataset es el radio estelar (R_s) medido en radio solar de cada objetivo. Para esta tarea se realiza crossmatch con 3 catálogos: TIC y Gaia DR2

¹⁷Los *Threshold Crossing Event* o TCE son eventos que corresponden a secuencias presentes en las curvas de luz que se asemejan a la señal de un exoplaneta.

utilizando el portal MAST, y CADARS¹⁸ utilizando el servicio VizieR¹⁹. Si el radio estelar está presente en TIC se utiliza ese valor, sino se busca en GAIA DR2 y finalmente en CADARS. Si no está presente en ninguno se realiza el siguiente procedimiento:

- Utilizar el servicio SIMBAD²⁰ para obtener los valores Vmag, Bmag, plx (parallax) y teff (Effective Temperature), si están disponibles.
- Seleccionar una Teff de SIMBAD, TIC o GAIA DR2. Si no está disponible, utilizar `pyasl.Ramirez2005().colorToTeff`²¹ de PyAstronomy, clase utilizada para calcular la Teff con Bmag – Vmag.
- Realizar crossmatch con Gaia DR3 usando el portal MAST para obtener valores de paralaje disponibles.
- Calcular paralaje en $[mas]$ ($1[mas] = 1/1000[arcsecond]$) utilizando los datos de SIMBAD, TIC o GAIA DR3.
- Utilizar los datos anteriores para calcular el radio estelar de cada estrella según se presenta en el Listing 3.2. Los resultados de las estrellas de interés están presentes en el Anexo

¹⁸Catalogue of Stellar Diameters (CADARS) (Pasinetti-Fracassini+ 2001)

¹⁹<http://vizier.cds.unistra.fr/>

²⁰<https://astroquery.readthedocs.io/en/latest/simbad/simbad.html>

²¹https://pyastronomy.readthedocs.io/en/latest/pyaslDoc/aslDoc/aslExt_1Doc/ramirez2005.html

```

1
2 # calculate absolute magnitude
3 def calculate_abs_mag(Vmag, mas):
4     distance = 1/mas
5     return Vmag - 5*np.log10(distance) + 5
6
7 # calculate stellar radius
8 def calculate_R(m, Teff):
9     # Sun Effective temperature: 5772 K
10    # Sun Absolute magnitude +4.83
11    L = 2.5**(4.83-m)
12    T = 5772/Teff
13    return (T**2)*(L**0.5)
14
15 m = calculate_abs_mag(Vmag, mas)
16 stellar_radius = calculate_R(m, Teff)

```

Listing 3.2: Código utilizado para calcular radios estelares.

Según los datos de exoplanetas descubiertos en NASA Exoplanet Archive con el método de tránsito, el menor radio exoplanetario es 0.026 radio Júpiter y el mayor es 2.085, por lo que se agrega una restricción inicial para el radio planetario $\in [0,02, 2,1]$ radio Júpiter. Realizando pruebas preliminares sobre esta restricción se obtienen malos resultados en las vistas local y global, donde no se logra apreciar visualmente curvas similares a las presentadas por [Yu *et al.*, 2019] y es difícil diferenciar visualmente los ejemplos PC y NOT_PC. Para buscar la causa de estos resultados se verifica lo siguiente:

- Podría ser que la duración de tránsito es muy corta en comparación a la resolución temporal de BRITE o que los tránsitos inyectados caigan fuera de las ventanas de observación.
- El método puede no dar buenos resultados dada la presencia de tendencias (*trends*) en las curvas concatenadas.
- Si hay mucha variabilidad (medida en desviación estándar de la magnitud de la curva concatenada), la inyección de tránsito no logra distinguirse lo suficiente para generar una vista local o global con un tránsito visualmente apreciable.
- El valor de $k = R_p/R_s$ puede ser demasiado pequeño para la precisión de BRITE. Al respecto es útil considerar que el valor $(R_p/R_s)^2$ se aproxima a la profundidad del tránsito. Mientras mayor sea la profundidad, más resalta el tránsito frente al ruido y variabilidad de los datos.

En consecuencia se imponen las siguientes restricciones y modificaciones.

1. Considerando un periodo orbital de los satélites cercano a 100 minutos y ventanas de observación de aproximadamente 20 minutos, la duración del tránsito se restringe a

un mínimo de 1 hora. Adicionalmente se restringe que la duración de un tránsito sea mayor al periodo del tránsito.

2. Realizar un *detrend* con el paquete Wotan, tomando como referencia la configuración usada por [Krishnamurthy *et al.*, 2021] en la búsqueda de tránsitos alrededor de Alpha Centauri A y B usando datos de la misión ASTERIA. Esto es un filtro *biweight* y ventana de 0,5 días.
3. Se calcula el menor radio estelar y mayor radio planetario tal que la profundidad $(Rp/Rs)^2$ sea mayor a 0.001 y que permita una variedad razonable de radios planetarios. Se decide utilizar la restricción de radio estelar menor a 3,08 radio Solar y radio planetario mayor a 0.95 y menor a 2.1 radio Júpiter.
4. Una mínima cantidad de puntos temporales durante todos los tránsitos de la curva. Experimentalmente se obtienen buenos resultados con un mínimo de 700 puntos.

El número de curvas concatenadas que cumplen la restricción (3) es 195, con un total de 137 objetivos diferentes, los cuales están presentes en las Tablas 6.1 y 6.2 en el ANEXO 6.3 RADIOS ESTELARES.

3.2.2. INYECCIÓN

Por cada curva concatenada que cumple las restricciones descritas en el subcapítulo anterior se intenta generar los ejemplos PC y NOT_PC siguiendo la proporción 16,03 % y 83,96 % NOT_PC, respectivamente. El número final de ejemplos puede variar debido a que durante la inyección de tránsitos, algunas combinaciones de parámetros no producen el tránsito (detectado como duración nula del tiempo de tránsito). Por cada curva se intenta un máximo de 50 veces la inyección con distintos parámetros hasta obtener un tránsito válido. En caso contrario, se descarta la creación de los ejemplos PC respectivos y se continúa con la creación de ejemplos NOT_PC de dicha curva.

Una curva con tránsitos se modela utilizando el modelo cuadrático del paquete PyTransit, el cual utiliza de entrada el valor de $k = Rp/Rs$ (radio planetario dividido por radio estelar), 2 coeficientes de *limb darkening* (μ_1, μ_2), tiempo t_0 , periodo p en días, a (semi-eje mayor dividido por radio estelar), inclinación i en radianes, excentricidad e , argumento del periastro w y un arreglo de tiempos de exposición donde se evalúa el modelo. Los valores a utilizar se obtienen aleatoriamente con una distribución uniforme en los rangos descritos en la Tabla 3.3, exceptuando los parámetros de *limb darkening* donde se utilizan distribuciones gaussianas. El valor t_0 corresponde al tiempo donde el tránsito se encuentra centrado²².

Para determinar los índices temporales que corresponden a la caída de brillo producto del tránsito se consideran los tiempos dentro de los rangos $[(t_0 \pm p \cdot j) - 0,5 \cdot$

²²Nota: debido a su periodicidad, el tránsito también esta centrado en los tiempos $t_0 \pm p \cdot j, j \in \{1, 2, 3, \dots\}$

Parameter	Values
p	1-5 [days]
a	10-30
R_p	0.95 - 2.1 [R _{jup}]
i	88 - 90 [°]
μ_1	$\mathcal{N}(\mu, \sigma^2)$, $\mu = 0,3$, $\sigma = 0,05$
μ_2	$\mathcal{N}(\mu, \sigma^2)$, $\mu = 0,1$, $\sigma = 0,02$
e	0.0
w	0.0
t_0	Random time in curve

Tabla 3.3: Rango de valores usados en la inyección de tránsito. Fuente: Elaboración propia.

$transit_duration, (t_0 \pm p \cdot j) + 0,5 \cdot transit_duration]$, $j \in \{0, 1, 2, 3, \dots\}$. Estos índices son remplazados en la curva BRITE concatenada por los valores de *flux* correspondientes a los índices de la curva con tránsitos modelada. La curva resultante es utilizada para crear las 3 representaciones. La duración del tránsito se obtiene con la función `pyasl.transitDuration`²³ de PyAstronomy. Un pseudocódigo con el procedimiento aplicado se encuentra en el Listing 3.3.

Una vez se genera una curva inyectada con periodo p , siguiendo el procedimiento descrito por [Yeh y Jiang, 2020] se calculan 2 periodos extra con la finalidad de crear 6 representaciones adicionales con periodos dos minutos más cortos y dos minutos más largo que p , esto es $p \pm 2/1440[days]$. Luego, por cada periodo se generan las 3 representaciones *phase fold*, *local view* y *global view*. De esta forma, si no se detecta un error durante el proceso²⁴ se obtienen 9 representaciones correspondientes a 3 ejemplos PC. Seis ejemplos de tránsitos inyectados y sus 3 representaciones se encuentran en la Figura 3.7.

Los ejemplos NOT_PC se generan de forma similar a los PC: aleatoriamente se obtienen parámetros de tránsito que cumplan las restricciones sobre una curva pero el tránsito no es inyectado, luego se crean las 3 representaciones *phase fold*, *local view* y *global view* considerando un periodo p , esta vez sin los 2 periodos adicionales.

²³<https://pyastronomy.readthedocs.io/en/latest/pyaslDoc/aslDoc/transitDuration.html>

²⁴Nota: es posible que se genere un *runtime error* durante la creación de la vista local o global, en tal caso se continúa con el siguiente periodo.

```

1
2 import numpy as np
3
4 def inject_transit(data, k, p, a, i, e, w, star_radius, t0):
5
6     # detrend using wotan
7     flatten_flux, trend = flatten(data["Time"], data["Magnitude"] + 1,
8         window_length=0.5, return_trend=True, method='biweight')
9
10    # generate curve with transits
11    flux_transit = generate_artificial_transit(data["Time"], k,
12        t0, p, a, i, e, w)
13
14    # since pytransit.QuadraticModel use 1 as base for flux values
15    # and the BRITE decorrelation use 0 as base values
16    flux_transit -= 1
17
18    if ( all values in flux_transit are infinite ): # handle bad case
19        # return empty flux
20
21    '''
22    sma:The semi-major axis in AU.
23    planet_radius: The planetary radius in Jovian radii.
24    star_radius: The stellar radius in solar radii.
25    '''
26    transit_duration = calculate_transit_duration(sma = sma,
27        rp = planet_radius, rs = star_radius, i = i, period = p)
28
29    times = data["Time"]
30    index_to_replace = []
31
32    t_i = t0
33
34    delta = 0.5*transit_duration
35
36    while(t_i <= max(times) ): # time > t0
37
38        # get time indexs that satisfy the condition:
39        [( data["Time"] >= (t_i - delta) ) &
40            ( data["Time"] <= (t_i + delta))]
41
42        # add indexs to index_to_replace
43
44        t_i += p # update t_i
45
46    t_i = t0
47    while(t_i >= min(times)): # time <= t0
48        # get time indexs that satisfy the condition:
49        [( data["Time"] >= (t_i - delta) ) &
50            ( data["Time"] <= (t_i + delta))]
51
52        # add indexs to index_to_replace
53

```

```
54     t_i -= p # update t_i
55
56     flux = flatten_flux
57     for i, idx in enumerate(index_to_replace):
58         flux[idx] = flux[idx] - np.absolute(flux_transit[idx])
59
60     return times, flux, t0, transit_duration
```

Listing 3.3: Pseudocódigo creado para inyectar tránsitos sintéticos.

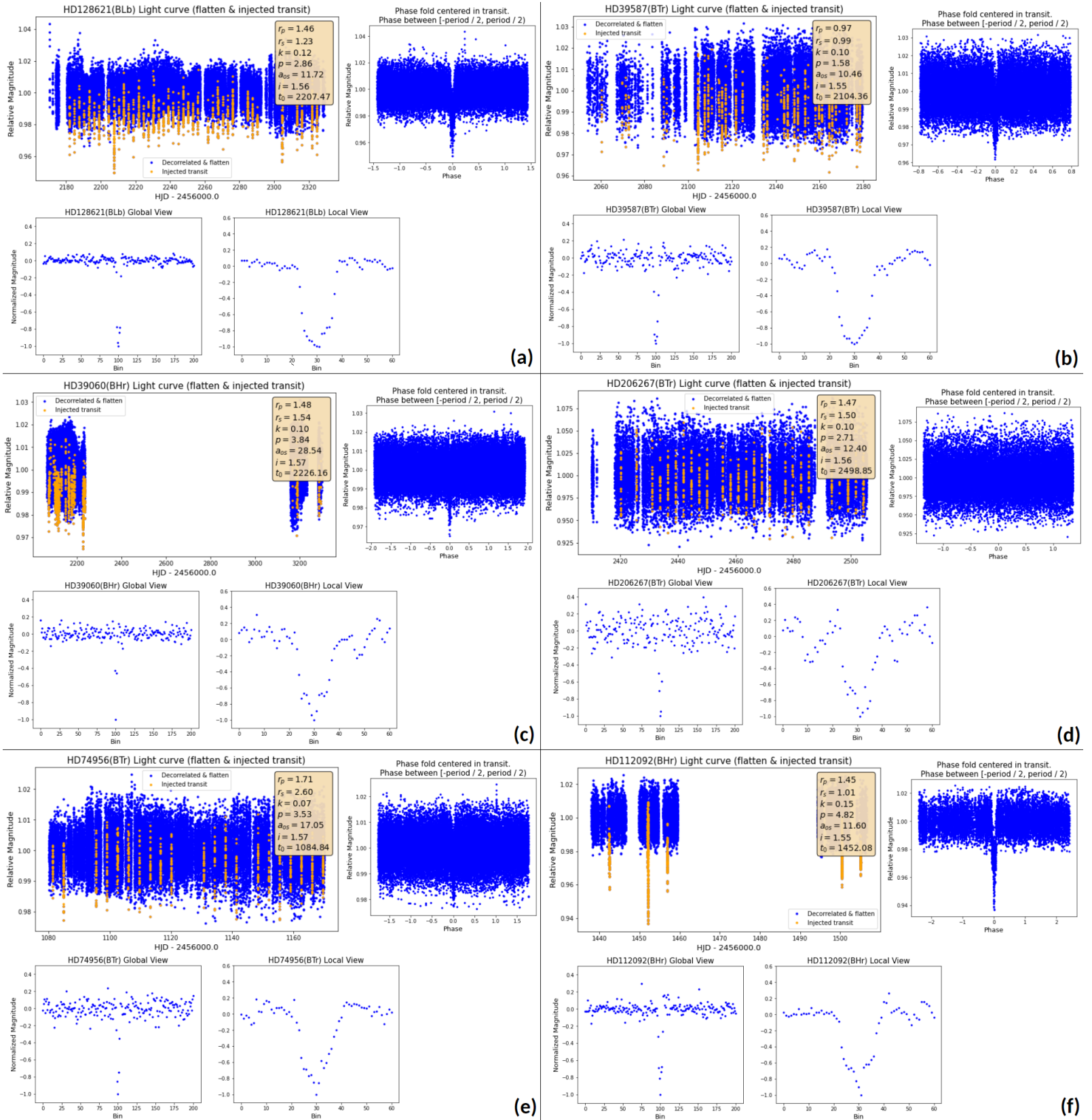


Figura 3.7: Ejemplos de tránsitos generados y sus representaciones Fold, Local y Global View. Los puntos color naranja corresponden al tránsito injectado. Fuente: Elaboración propia.

La creación del *dataset* sobre las 195 curvas concatenadas resulta en 2404 ejemplos PC y 9616 ejemplos NOT_PC. Se almacena las representaciones en archivos .npy de Numpy y todos los parámetros utilizados para la inyección del tránsito en un archivo .csv.

El siguiente paso es validar los ejemplos generados con la finalidad de seleccionar un subconjunto del total para entrenar y evaluar los modelos de detección tal que estos ejemplos cumplan un criterio de calidad. Este filtro es detallado en el SubCapítulo 4.2. VALIDAR EL DATASET. Se seleccionan los 597 ejemplos PC que cumplen el criterio de calidad y aleatoriamente se escogen 3126 ejemplos NOT_PC para obtener la proporción de 16.04 % y 83.96 %.

A continuación se realiza la separación aleatoria de los ejemplos seleccionados en dos subconjunto: entrenamiento/validación y test. Se utiliza un 10 % del total de ejemplos para el *testing*, por lo que los subconjuntos poseen 3350 y 373 ejemplos, respectivamente. En este punto el *dataset* esta listo para entrenar modelos de Deep Learning.

3.3. APLICACIÓN DE MODELOS

3.3.1. SELECCIÓN Y ENTRENAMIENTO

Ya creado el *dataset*, la tarea de detección se realiza con modelos Deep Learning, en particular implementando las arquitecturas convolucionales (CNN) utilizadas por [Yeh y Jiang, 2020] y [Yu et al., 2019] (AstroNet Triage), ambas descritas en el SubCapítulo 2.3.5. APLICACIONES. Los códigos con ambos modelos son proporcionados por los autores, usando librerías Keras y Tensorflow, respectivamente.

Antes de aplicar modelos se realiza un breve análisis exploratorio de los subconjuntos de entrenamiento y validación. En la Figura 3.8 se presentan histogramas de distintos parámetros utilizados en la generación de ejemplos. Se aprecia que los radios estelares tienden a ser menores en los ejemplos PC respecto a los NOT_PC. Respecto al radio planetario, en los ejemplos PC se presenta un sesgo negativo con una moda cercana a $1.95 [R_{jup}]$, mientras que para NOT_PC estos radios son uniformes. El comportamiento de estos dos parámetros se puede explicar considerando que $(R_p/R_s)^2$ aproxima la profundidad del tránsito, donde una mayor profundidad permite ejemplos con un tránsito apreciable frente al ruido de los datos. Por otro lado, el sesgo positivo del parámetro (a) es debido a que un mayor valor se obtiene si el exoplaneta se encuentra más alejado de su estrella, lo que produce un tránsito con menor profundidad.

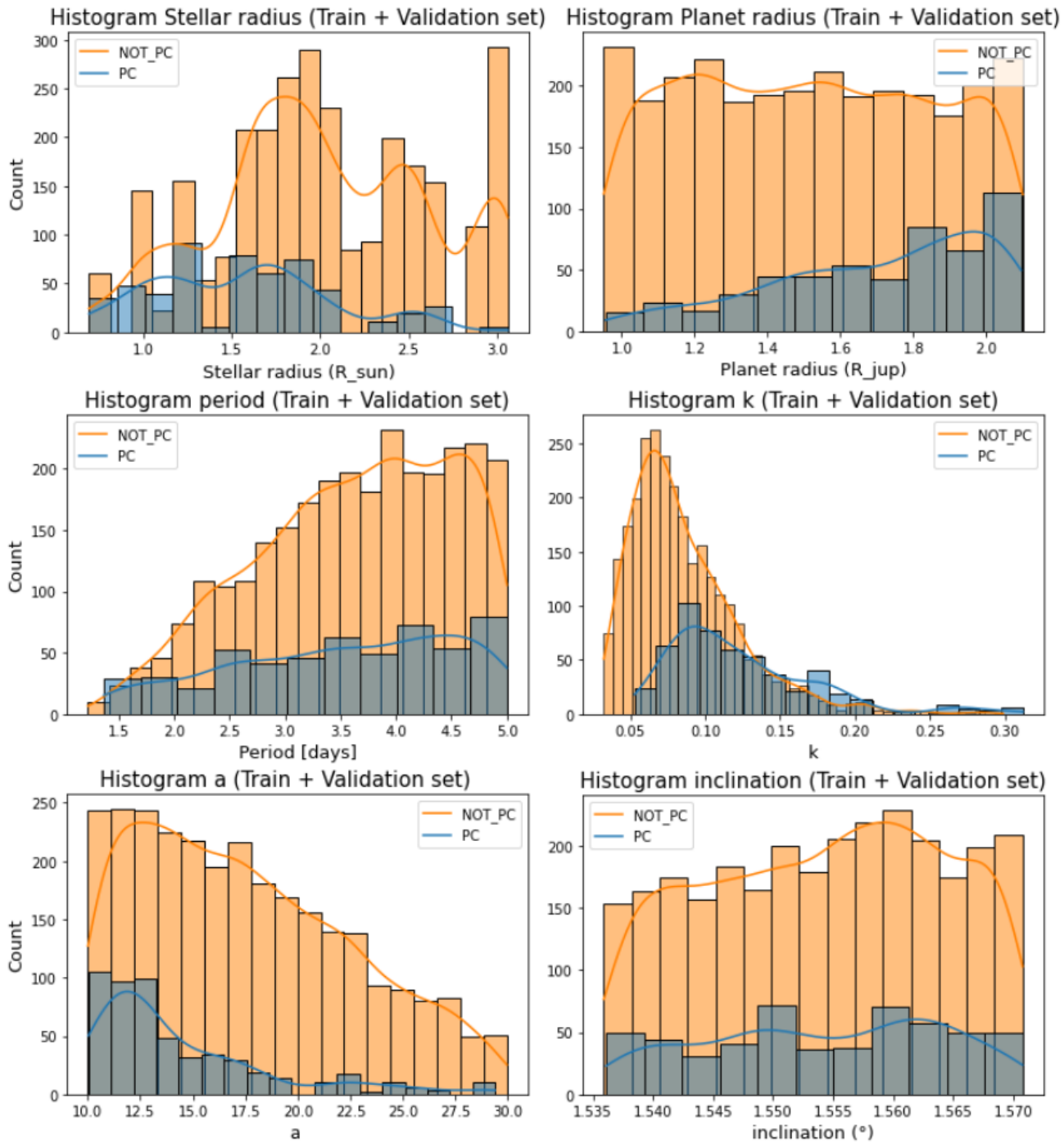


Figura 3.8: Histogramas con los valores de los parámetros en los subconjuntos de entrenamiento y validación. Fuente: Elaboración propia.

Por conveniencia, el modelo AstroNet-Triage se implementa en Keras y se modifica el tamaño de entrada del modelo CNN de [Yeh y Jiang, 2020] para un input de 201 *features* (*global view*). Para evaluar los distintos modelos se realiza un K-Folds Cross-Validation con 10 grupos

para determinar el número de epoch a utilizar. Los siguientes hiperparámetros y configuraciones son fijos para todos los modelos:

- 3015 ejemplos de entrenamiento y 335 ejemplos validación por cada fold.
- Loss Binary CrossEntropy.
- Optimizador Adam con learning rate 1e-05.
- Callback EarlyStopping, con paciencia 8, monitoreando validation loss.
- Máximo número de epoch: 292.
- Número de ejemplos por actualización de gradiente: 70.

El detalle del desempeño los modelos se encuentra en el SubCapítulo 4.3 VALIDACIÓN DE MODELOS. **El desempeño global es mejor para el modelo Astronet-Triage**, por lo que se decide utilizar como modelo final dicha arquitectura e hiperparámetros. El valor final de *epoch* se fija en 46 al ser el valor promedio de los *epochs* resultantes de los 10 folds.

El **modelo final**, llamado `astronet_46`, es entrenado con los 3350 ejemplos disponibles del train set y validation set, sin el paso de validación. La *loss* resultante por *epoch* es presentada en la Figura 3.9. Para medir el rendimiento del modelo en datos independientes del entrenamiento se predicen las etiquetas de los 373 ejemplos del Test Set. En la Figura 3.10 se encuentra la matriz de confusión y las métricas Precisión, Recall y AUC-ROC considerando un umbral de 0.5, es decir, si el modelo predice un valor mayor o igual a 0.5 se considera PC y en caso contrario NOT_PC. Los resultados cercanos a 1.0 en todas las métricas sugiere un buen rendimiento del modelo sobre datos futuros en la tarea de detección de posibles exoplanetas.

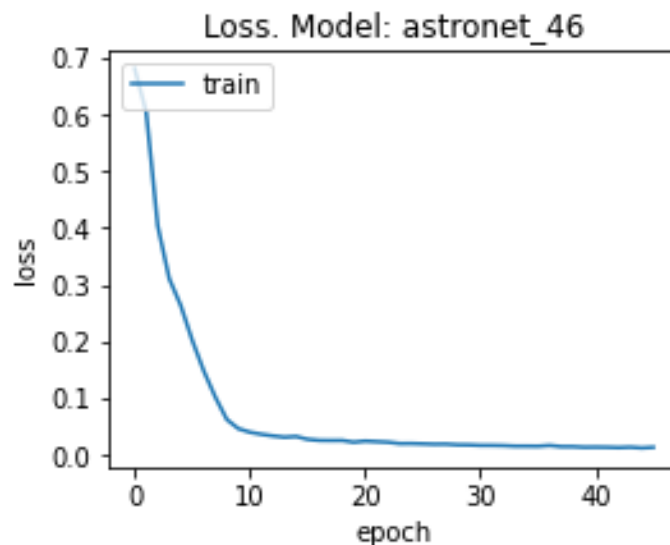


Figura 3.9: Loss por *epoch* del modelo final (`astronet_46`). Fuente: Elaboración propia.

Confusion Matrix astronnet_46 with Test Set

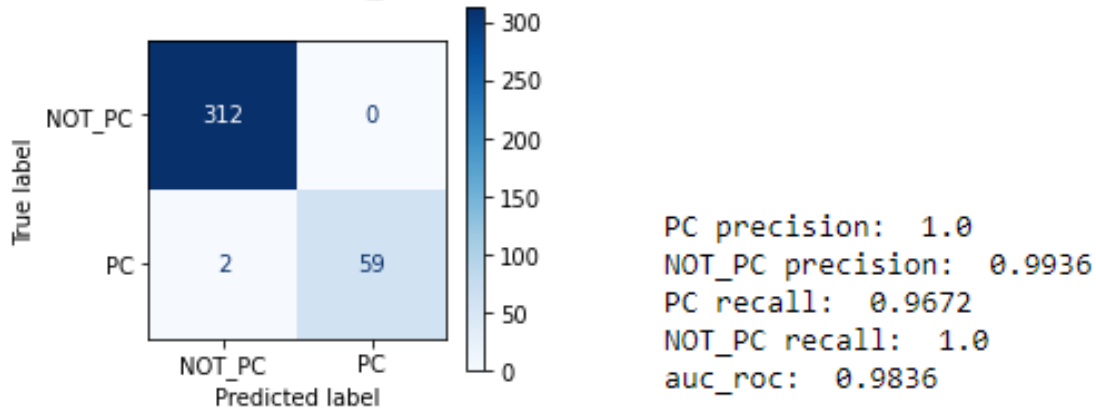


Figura 3.10: Matriz de confusión y métricas del modelo final (astronnet_46) sobre el Testing Set. Fuente: Elaboración propia.

3.3.2. BÚSQUEDA DE POSIBLES EXOPLANETAS

El último paso es buscar posibles exoplanetas en los datos pre-procesados de BRITE. Debido al alcance temporal de esta memoria, la búsqueda se enfoca en las estrellas objetivo presentes en los datos que poseen exoplanetas ya confirmados en NASA Exoplanet Archive a la fecha 15/06/2022 y que cumplan la restricción de radio menor a 3.08 radios solares. Los cuatro objetivos que cumplen estas condiciones son **HD 039060**, **HD 022049**, **HD 217014** y **HD 218396**, cuyos radios son 1.54416, 0.755368, 1.17561 y 1.49338 [R_{sun}], respectivamente. Otros resultados interesantes de otros objetivos también se presentan a continuación.

La búsqueda se realiza definiendo las estrellas objetivos, rango de periodos a probar (máximo y mínimo), tiempo entre periodos, y valor de predicción mínimo (score) para registrar el caso como PC. El número de periodos a probar por objetivo se distribuye uniformemente entre $[min_period, max_period]$, con un valor temporal de $max_difference$ entre periodos de prueba consecutivos. Por cada periodo p se prueban distintos valores de $epoch$ de prueba, los cuales se distribuyen uniformemente entre $[min_time, min_time + p]$, con un valor alrededor de 200 minutos entre $epochs$ consecutivos y min_time es el menor valor temporal de la curva de luz. Además, por cada periodo de prueba, aleatoriamente se escoge una inclinación, radio planetario y semi-eje mayor dividido por radio estelar, dentro de los rangos descritos en la generación de ejemplos.

Para los objetivos **HD 039060**, **HD 022049**, **HD 217014** y **HD 218396** se realizan dos búsquedas, ambas con un umbral de predicción de 0.8. La primera búsqueda resultó en 75 casos con $score > 0,8$ en un tiempo de ejecución de 03:35 horas con la siguiente configuración:

- $min_period = 1$ día.

- `max_period` = 3 días.
- `max_difference` = 2.5 minutos.
- Número de periodos de prueba = 1152.

La segunda búsqueda resultó en 577 casos con `score` $> 0,8$ en un tiempo de ejecución de 14:32 horas con la siguiente configuración:

- `min_period` = 5 días.
- `max_period` = 10 días.
- `max_difference` = 5 minutos.
- Número de periodos de prueba = 1440.

En ambos resultados se realiza una inspección visual de los casos para descartar clasificaciones donde claramente la señal no se semeja a un exoplaneta e identificar los casos donde la señal si se asemeja a un candidato. Adicionalmente se realizan otras búsquedas disminuyendo el umbral de predicción a 0.5 para otros objetivos.

En las Figuras 3.11 y 3.12 se presentan casos interesantes encontrados cuya forma visual se asemeja a un posible candidato. En la primera Figura, el caso (c2) es el (c1) con el *time epoch* modificado a mano para centrar el tránsito. Para el caso (d), la señal característica solo se encuentra en la *Global View*. Una inspección más detallada de estos candidatos debe ser realizada por un experto, tarea designada a trabajo futuro.

Durante la inspección visual se descubren señales que se asemejan a una estrella variable, por lo que el modelo clasifica erróneamente esta variabilidad como exoplaneta. Casos notables se presentan en la Figura 3.13. HD 218396 es un candidato que hay que tener especial cuidado porque se detectó una señal similar a estrella variable.

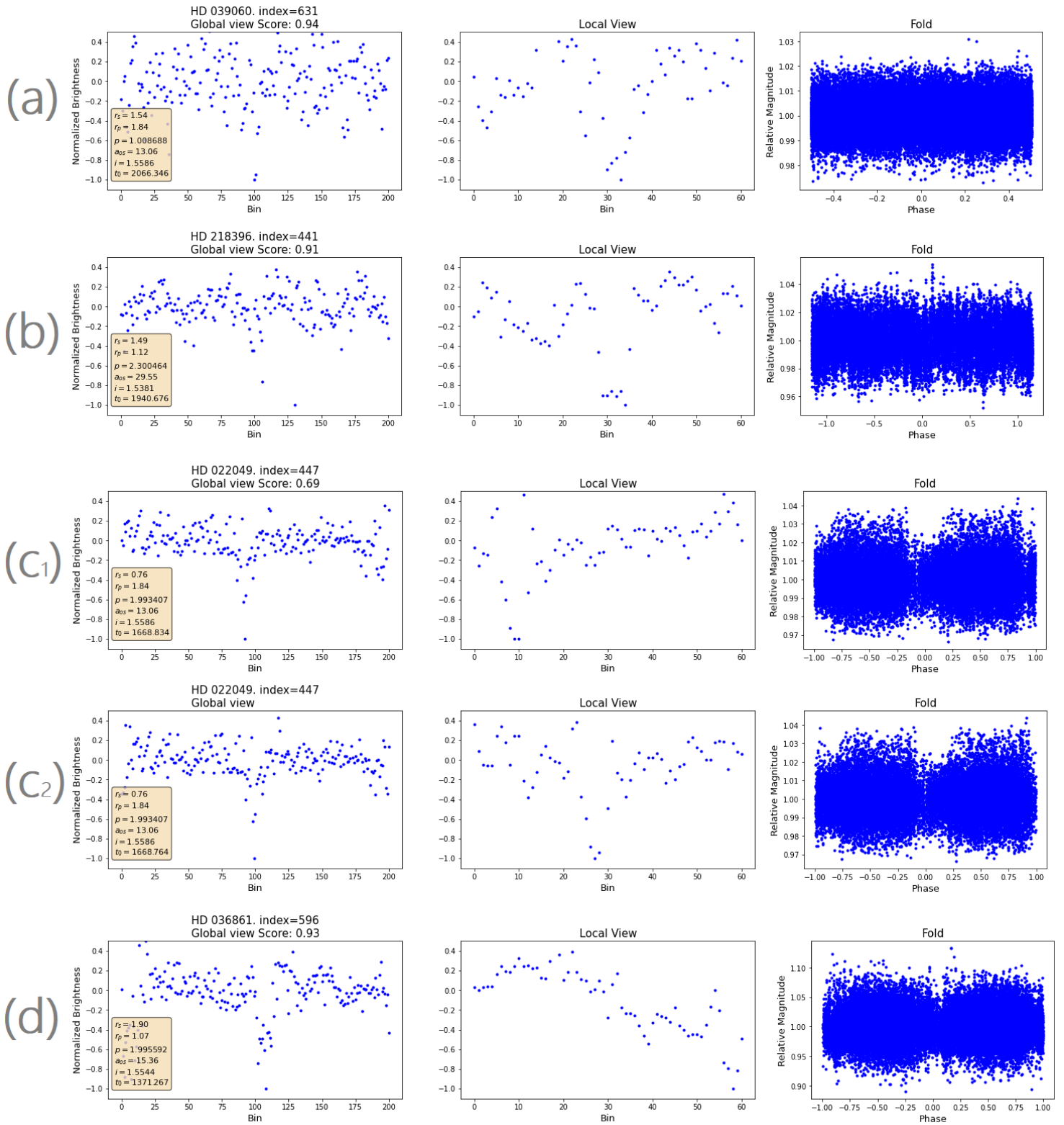


Figura 3.11: Casos encontrados con señales similares a un exoplaneta en los datos de BRITE, junto a los parámetros predichos. Fuente: Elaboración propia.

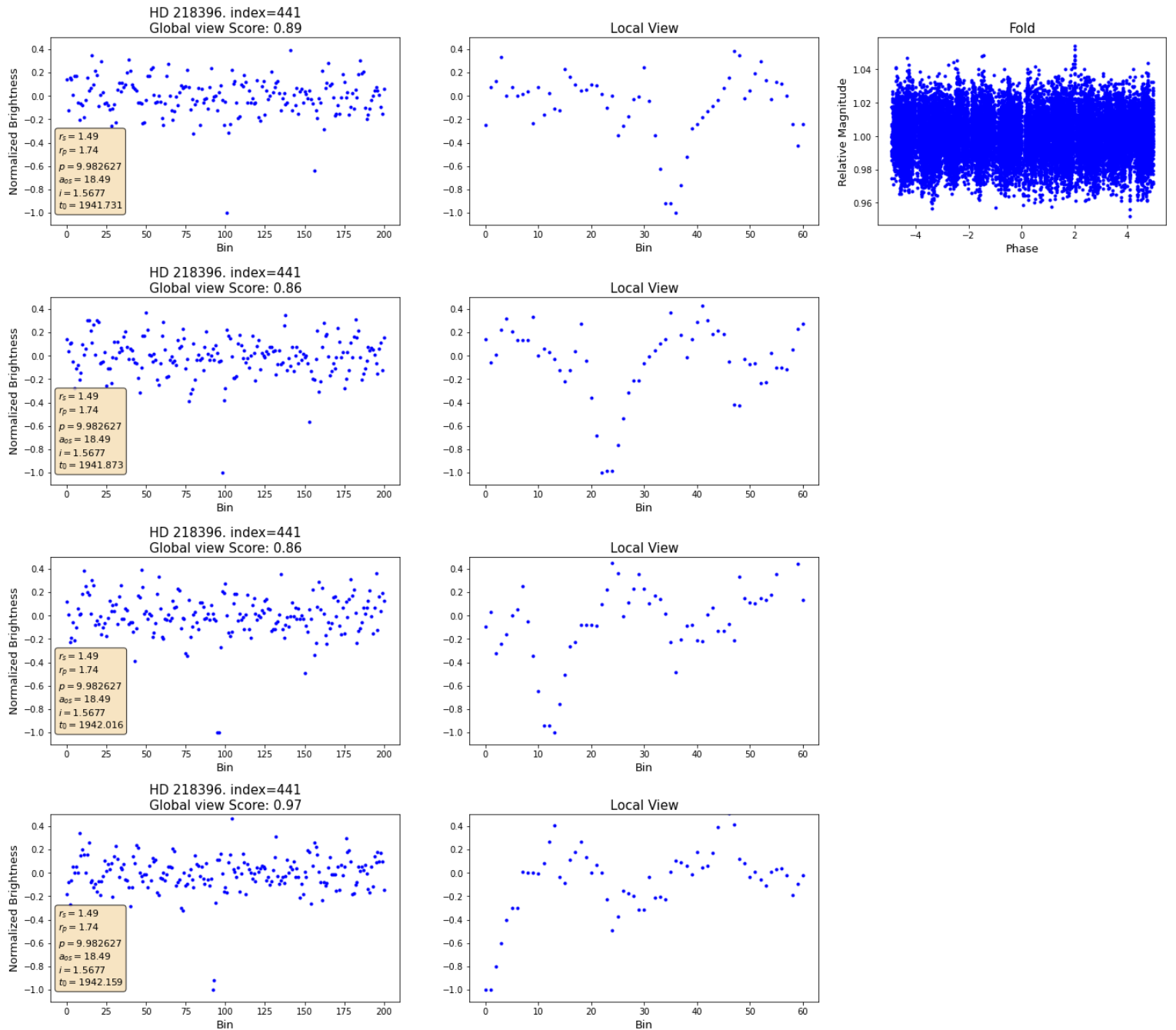


Figura 3.12: Casos encontrados cuyas señales son similares a un exoplaneta detectadas en HD 218396 en *time epochs* cercanos. Fuente: Elaboración propia.

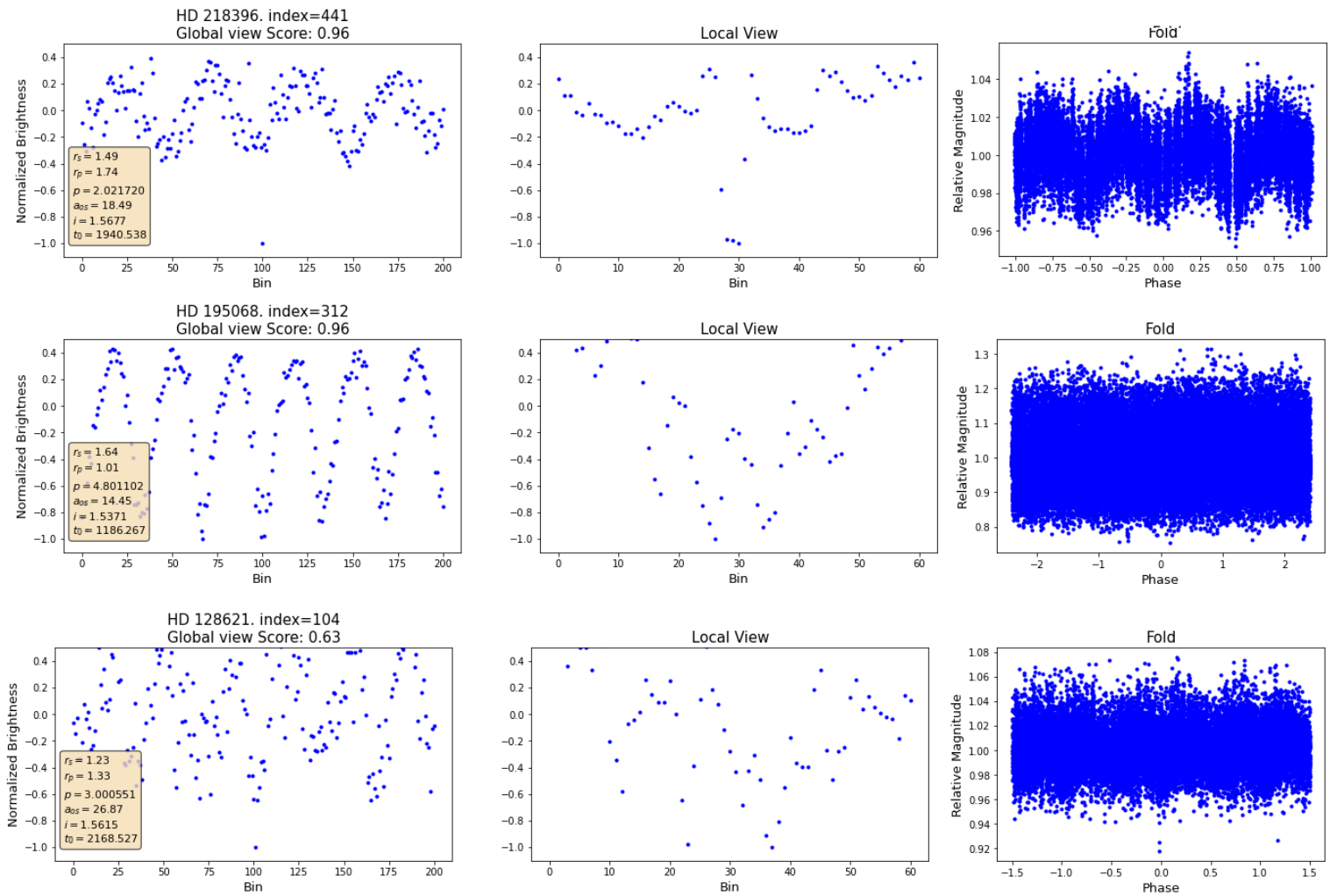


Figura 3.13: Casos encontrados cuya señal se asemeja a estrellas variables. Fuente: Elaboración propia.

CAPÍTULO 4

VALIDACIÓN DE LA SOLUCIÓN

4.1. VALIDAR EL PREPROCESAMIENTO

Para validar que el pre-procesamiento descrito en el Subcapítulo 3.1.3 PREPROCESAMIENTO es correcto, se recrea el análisis de [Sódor y Bognár, 2020]. En este trabajo los autores identifican frecuencias de pulsación significativas en la estrella HR 8799 utilizando datos BRITE, comparándolas con las detectadas en datos de MOST²⁵. En el trabajo presentan un espectro de amplitud de Fourier y los componentes de frecuencia extraídos de la fotometría BRITE. Si el pre-procesamiento es apropiado, se espera detectar las mismas frecuencias.

Se utilizan los datos públicos de HR 8799 para tiempos mayores a 1980 [HJD - 2456000] y se obtienen las frecuencias utilizando un periodograma Lomb-Scargle sobre los datos. En la Figura 4.1 se presenta en la parte superior el gráfico original de [Sódor y Bognár, 2020] y en la parte inferior el periodograma Lomb-Scargle generado. El valor de $f_1 = 1,97984[1/day]$ se obtiene del trabajo de [Sódor *et al.*, 2014]. Visualmente se aprecia que las frecuencias de mayor amplitud coinciden perfectamente al igual que la frecuencia menor f_4 y en menor medida la frecuencia $f_1 \cdot \frac{14}{9}$, por otro lado, la frecuencia f_5 no coincide. Es posible que sea debido a que en el trabajo original se utiliza mayor cantidad de puntos temporales. Este resultado corrobora una implementación apropiada para el pre-procesamiento.

²⁵Microvariability and Oscillations in STars

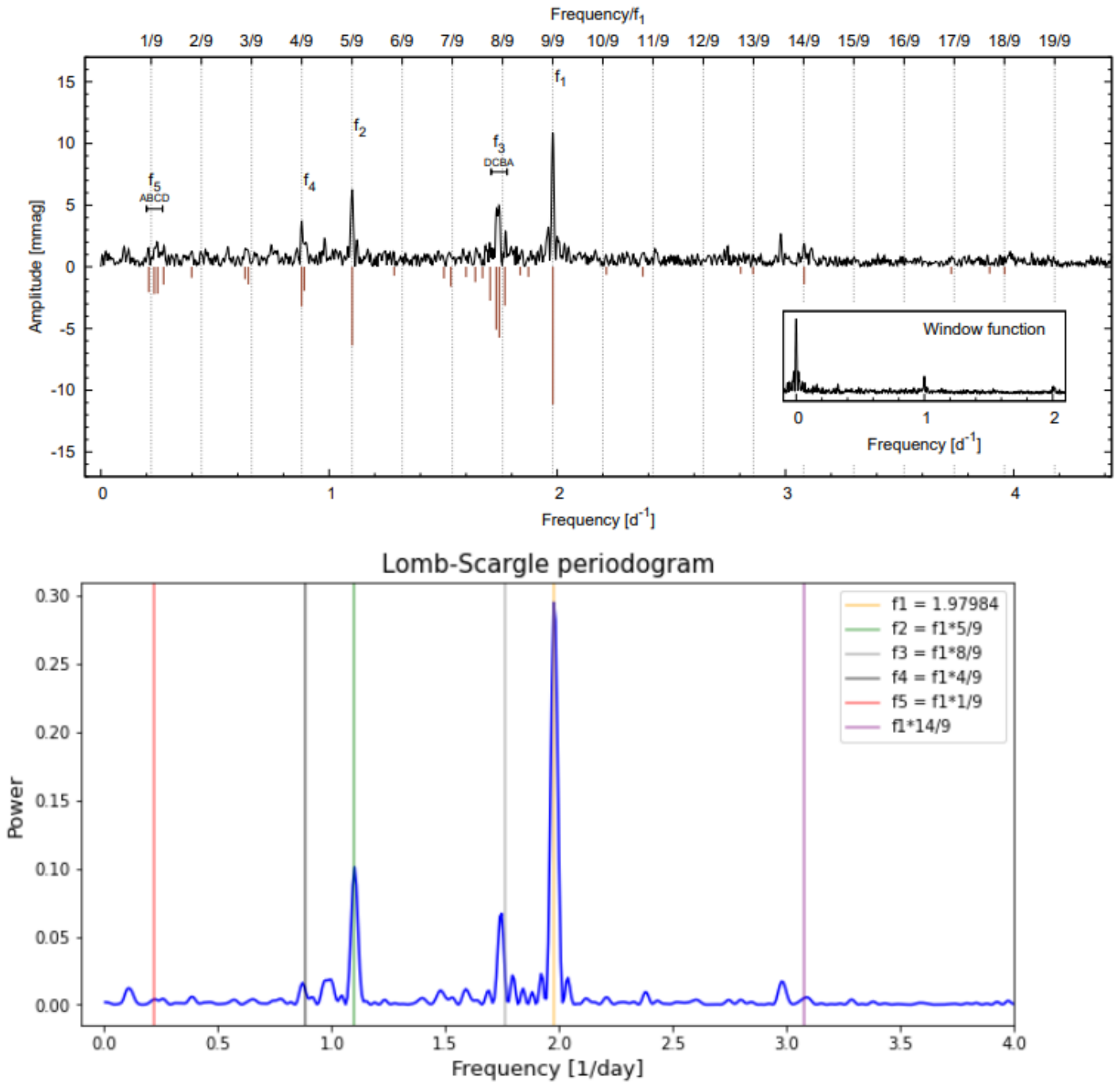


Figura 4.1: (superior) Espectro de amplitud de Fourier de HR 8799 calculado por [Sódor y Bognár, 2020]. (inferior) Periodograma Lomb-Scargle generado con los datos pre-procesados. Fuente: [Sódor y Bognár, 2020] y elaboración propia.

4.2. VALIDAR EL DATASET

Generado el dataset se verifica su calidad y se filtran los ejemplos que no cumplan un criterio de calidad. Para validar el dataset se utiliza la arquitectura AstroNet-Triage propuesto por [Yu *et al.*, 2019]. Por conveniencia el modelo AstroNet-Triage es re-implementado en Keras y en consecuencia se vuelve a entrenar. El trabajo original proporciona 13201 ejemplos para entrenamiento almacenados en 8 *tfrecord*²⁶ y 1657 ejemplos de validación almacenados en 1 archivo *tfrecord*. Originalmente esta configurado un número de *steps* de 14000, entonces considerando 13201 ejemplos y *batch size* de 64, el número de *steps* por *epoch* es $\text{ceil}(\frac{13201[\text{samples}/\text{epoch}]}{64[\text{samples}/\text{step}]}) = 207$, por lo tanto, el número de *epoch* a usar es $\text{ceil}(\frac{14000[\text{steps}]}{207[\text{steps}/\text{epoch}]}) = 68$. Se compila el con *loss binary_crossentropy* y optimizador Adam con *learning_rate* de $1e-05$. A diferencia del trabajo original, no se realiza "*model avering*". Métricas resultantes del entrenamiento y validación se encuentran en la Figura 4.2. Se puede apreciar que se puede mejorar la implementación ya que se se presenta *overfitting* desde el *epoch* ~ 40 y el *accuracy* podría ser mayor, sin embargo, se decide continuar el trabajo usando este modelo entrenado ya que: (1) se corrobora una implementación apropiada en Keras y (2) este modelo se utiliza para validar los ejemplos generados y no la tarea de clasificación final.

El modelo implementado en Keras recibe de entrada un ejemplo del dataset generado (representación *local* y *global view*) y entrega una predicción entre 0 y 1. Se considera un umbral de 0.5, donde un valor ≥ 0.5 se considera una predicción PC, mientras que un valor < 0.5 se considera NOT_PC. El modelo es capaz de asignar un valor < 0.5 el 100 % los ejemplos generados etiquetados como NOT_PC, mientras que solo en 360 de los ejemplos etiquetados como PC obtienen un valor > 0.5 , lo que corresponde a un $\sim 20\%$. Al no ser un 100 %, el *dataset* final solo contiene ejemplos PC generados que cumplan el siguiente criterio de calidad:

1. Obtenga una predicción > 0.5 , o
2. Al menos 1 de los ejemplos generados a partir del mismo periodo p cumple el criterio anterior, i.e. pertenece al grupo de ejemplos generados con el periodo p y $p \pm 2/1440[\text{days}]$. En tal caso todos los ejemplos PC del grupo son considerados para el *dataset* final.

El número de ejemplos PC que cumplen el criterio de calidad es de 597. Finalmente, el *dataset* final validado consiste en los 597 ejemplos PC y 3126 ejemplos NOT_PC elegidos aleatoriamente, de esta forma se obtiene la proporción 16.04 % y 83.96 %.

²⁶*tfrecord* es un formato de almacenamiento de secuencias de registros binarios.

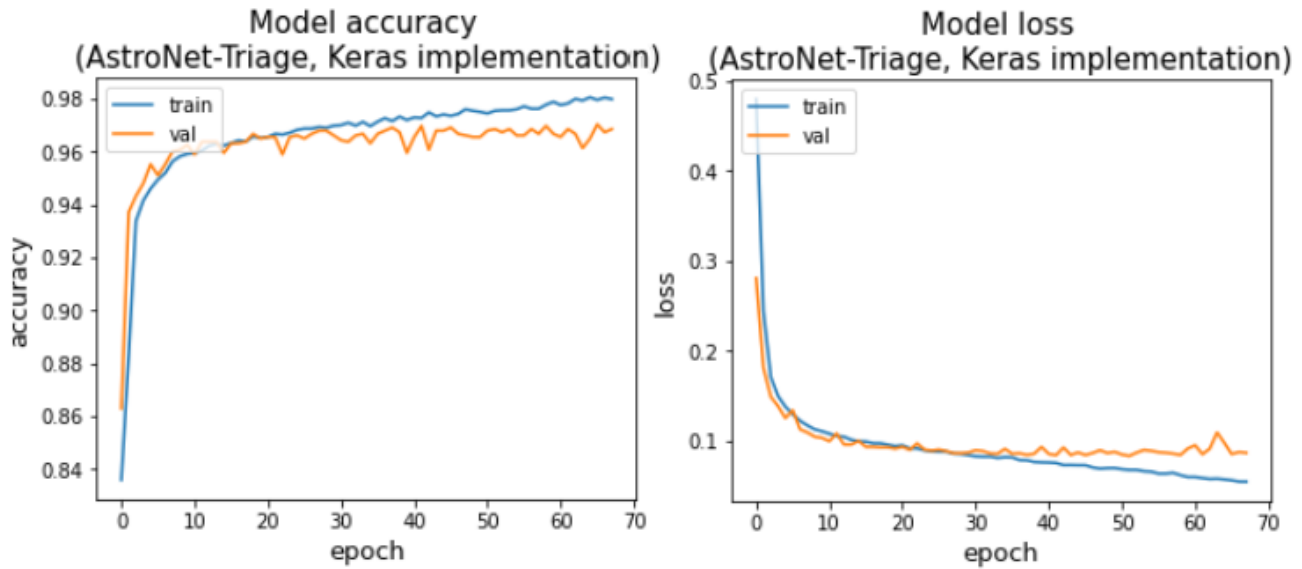


Figura 4.2: Métricas del entrenamiento y evaluación del modelo AstroNet-Triage implementado en Keras. Fuente: Elaboración propia.

4.3. VALIDACIÓN DE MODELOS

El propósito es validar los modelos implementados a través de la obtención de valores altos en métricas de evaluación, lo que permite adicionalmente seleccionar el modelo a utilizar para la tarea de detección y verificar un buen desempeño sobre datos futuros. Para ello se calculan las métricas Precisión, Recall y AUC-ROC sobre los conjuntos de validación para los dos modelos descritos en el SubCapítulo 3.3.1. SELECCIÓN Y ENTRENAMIENTO.

En las Figuras 4.3 y 4.3 se encuentran las matrices de confusión para los 10 *folds* del *K-fold Cross Validation*, junto a las métricas promedio. Todas las métricas obtienen un valor superior a 86 %, llegando al 99 % en algunas métricas, por lo que verifica la calidad de los modelos.

Se puede observar que el desempeño global es mejor para el modelo Astronet-Triage en todas las métricas, por lo que se utiliza dicha arquitectura como modelo final.

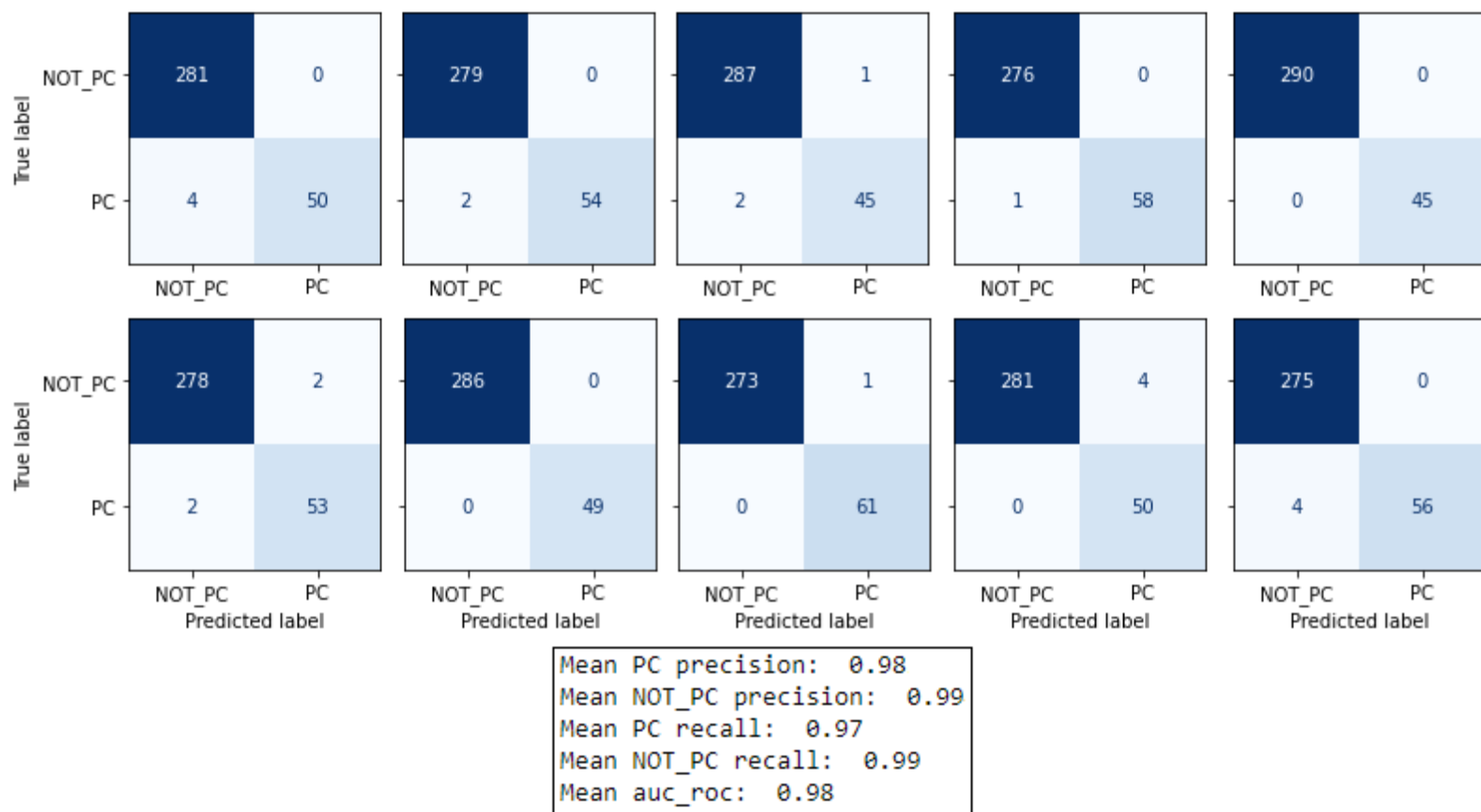


Figura 4.3: Matriz de confusión y métricas del modelo basado en AstroNet sobre conjunto de validación. Fuente: Elaboración propia.

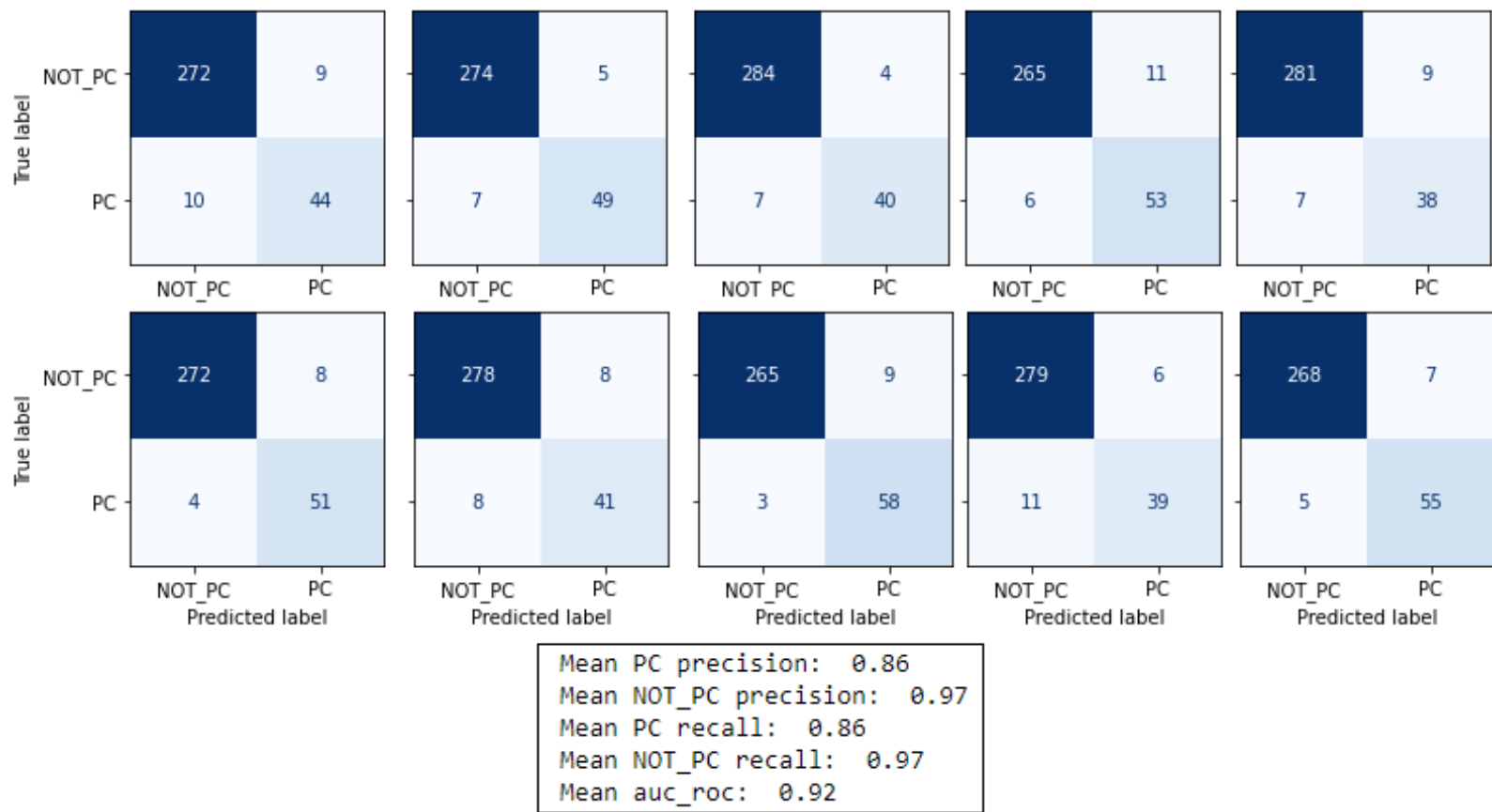


Figura 4.4: Matriz de confusión y métricas del modelo basado en [Yeh y Jiang, 2020] sobre conjunto de validación. Fuente: Elaboración propia.

CAPÍTULO 5

CONCLUSIONES

En esta memoria se estudió el problema de detección de exoplanetas utilizando tecnología nanosatelital, en particular se analizó la tarea de detección de posibles exoplanetas utilizando modelos de Deep Learning sobre los datos de la misión BRITE. Para cumplir este objetivo general se plantearon y abordaron los objetivos específicos de la siguiente forma:

- Se investigaron técnicas de representación de curvas de luz de tránsitos. Para ello se describe el fenómeno y detección del tránsito exoplanetario, junto a la revisión de métodos de análisis de curvas de luz como los periodogramas Lomb–Scargle y la técnica *detrending* para remover tendencias. Sobre las representaciones, se describe la representación en curvas *phase-folded* y las representaciones Global View y Local View utilizadas por [Yu *et al.*, 2019] en la búsqueda de exoplanetas en datos de TESS.
- Se seleccionaron las representaciones Global View y Local View como entrada de los modelos de detección. Ambas representaciones son vectores 1D de largo 201 y 61, respectivamente. Para generar curvas de luz con tránsito de forma sintética se investiga el método cuadrático de Mandel y Agol, el cual modela un tránsito con características de la estrella como su radio y coeficientes de *limb darkening*, y las propiedades del exoplaneta como su radio y parámetros orbitales.
- Se compararon distintos catálogos y *surveys* relacionados a tránsitos exoplanetarios a través de una revisión del contexto de sus misiones, características de sus instrumentos, captura de datos, disponibilidad de datos y número de descubrimientos. Las misiones detalladas son Kepler, su extensión K2 y TESS. Adicionalmente se hace una revisión de las misiones astronómicas con nanosatélites, destacando la constelación de 5 nanosatélites BRITE y el CubeSat ASTERIA.
 - Las curvas de luz producidas por el proyecto BRITE son la principal fuente de datos de esta memoria, por lo que se realiza una revisión detallada de sus características, haciendo especial enfoque en la estrategia de software "*chopping*", usada para mitigar el efecto de *hot pixels*, apuntamiento e ineficiencia en la transferencia de carga. Cabe destacar que, respecto a las otras misiones nombradas, en términos de *pipeline* de datos y catálogos de interés, la misión TESS es de mayor relevancia con su TESS Input Catalog o TIC. También es de interés los parámetros para *detrend* utilizados por [Krishnamurthy *et al.*, 2021] en los tránsitos en Alpha Centauri A y B vistos con la misión ASTERIA.
- Ya que se utilizan modelos de Deep Learning, se dedica un subcapítulo a los conceptos y terminología aplicada a lo largo del texto, dedicada principalmente un lector que no conoce el tema en profundidad. Se destaca una descripción de la arquitectura Red

Neuronal Convolutacional (CNN), utilizadas por [Yu *et al.*, 2019] y [Yeh y Jiang, 2020] en la búsqueda de exoplanetas. Adicionalmente se describen las métricas evaluación Precisión, Recall, Accuracy y AUC-ROC en el contexto de detección.

- Antes de seleccionar modelos de Deep Learning para la tarea de detección, debido a que la misión BRITE no esta diseñada originalmente para la búsqueda de exoplanetas **se crea un dataset de curvas de luz con señales sintéticas de exoplanetas, inyectados sobre de los datos de BRITE, siendo este uno de los principales aportes de la memoria.** Lo que requirió:

- Pre-procesar los datos originales según lo recomendado por [Pigulski, 2018]. Un aporte es la implementación de un método naive para remover outlier por órbita y remover peores órbitas ya que los datos originales no indican qué dato corresponde a que orbita. Este método se puede aplicar de forma general a otras misiones espaciales.
- Adicionalmente se propone modificar el paso de decorrelación limitando la cantidad de iteraciones a 20 y se propone una heurística para no volver a corregir la correlación de un parámetro si la mejora respecto a la última iteración anterior del parámetro es menor al 5 %, agregandolo a una "lista tabú". Se verifica una apropiada implementación del pre-procesamiento al recrear el análisis de [Sódor y Bognár, 2020] sobre estrella HR 8799.
- El pre-procesado implementado logra obtener 3139 curvas de luz, las cuales son concatenadas si (i) corresponden a la misma estrella objetivo, y (ii) corresponden al mismo satélite, lo que resulta en 876 curvas de luz. A continuación se adaptó la estrategia de [Yeh y Jiang, 2020] para inyectar señales de tránsito sobre estas curvas.

Las mejoras respecto a dicha estrategia es que se trabaja con mayor cantidad de objetivos, se aplica el pre-procesamiento sugerido por el equipo de BRITE, se agrega un paso de *detrending* y restricciones, siendo la principal que la profundidad del tránsito, aproximada con $(R_p/R_s)^2$, sea mayor a 0.001, limitando el número de estrellas objetivo a las que tenga radio menor a 3.08 radios solares y exoplanetas con radio entre 0.95 y 2.1 radio Júpiter.

- Un paso necesario y aporte de esta memoria es recopilar y calcular los radios estelares de los objetivos.
- La creación del dataset resulta en 2404 ejemplos PC y 9616 ejemplos NOT_PC, los cuales son filtrados con un criterio de calidad utilizando predicciones del modelo AstroNet-Triage propuesto por [Yu *et al.*, 2019]. El número de ejemplos PC que cumplen el criterio de calidad es de 597, por lo que el dataset final validado consiste en los 597 ejemplos PC y 3126 ejemplos NOT_PC elegidos aleatoriamente, obteniendo una proporción 16.04 % y 83.96 %.
- Con ello se realizó una separación aleatoria de los ejemplos seleccionados en dos subconjunto: entrenamiento/validación y test. Se utiliza un 10 % del total de

ejemplos para el testing, por lo que los subconjuntos poseen 3350 y 373 ejemplos, respectivamente.

- Para seleccionar los modelos de Deep Learning a partir del dataset creado, se comparan dos arquitecturas CNN: AstroNet Triage [Yu *et al.*, 2019] y la propuesta en [Yeh y Jiang, 2020], modificando el input a 201 features (global view). Para evaluar los distintos modelos se realiza un K-Folds Cross-Validation con 10 grupos para determinar el número de epoch a utilizar, manteniendo fijo los siguientes hiperparámetros y configuraciones:
 - Loss Binary CrossEntropy.
 - Optimizador Adam con learning rate 1e-05.
 - EarlyStopping, con paciencia 8.
 - Máximo número de epoch de 292.
 - 3015 ejemplos de entrenamiento y 335 ejemplos validación por cada fold.
 - Número de ejemplos por actualización de gradiente: 70.

Todos los modelos obtienen valores de Precisión, Recall y AUC-ROC promedio con un valor superior a 86 %, lo que verifica calidad en los modelos. El desempeño global es mejor para el modelo AstroNet-Triage, obteniendo valores promedios superiores a 97 % en todas las métricas. En consecuencia la selección de modelo final es la arquitectura AstroNet-Triage con un valor de epoch fijo de 46.

- Para evaluar el modelo final se entrena con todos los datos disponibles de entrenamiento y validación, y se predicen las etiquetas del conjunto de *testing* con un umbral de 0.5, obteniendo 100 % precisión y 96.72 % *recall* para la clase PC, y AUC-ROC de 98.36 %. Esto indica un buen rendimiento del modelo sobre datos futuros en la tarea de detección.
- Finalmente se buscan posibles exoplanetas en los datos pre-procesados de BRITE, con especial interés en los objetivos HD 039060, HD 022049, HD 217014 y HD 218396, que poseen exoplanetas confirmados.
 - **Se encuentran señales similares a tránsitos exoplanetarios con su respectivo periodo y time epoch en los objetivos HD 039060, HD 218396, HD 022049 y HD 036861. Estas señales, junto a sus parámetros orbitales, deben ser inspeccionadas por un experto para confirmar su naturaleza, tarea designada a trabajo futuro.**
 - Se descubren señales que se asemejan a una estrella variable, por lo que el modelo clasifica erróneamente esta variabilidad como exoplaneta. De todas formas se reportan periodos de variabilidad en los objetivos HD 218396, HD 195068 y HD 128621.

5.1. TRABAJO FUTURO

Las distintas partes de este trabajo abren la puerta a los siguientes trabajos futuros:

- Optimizar el *pipeline* de pre-procesamiento implementado para los datos de BRITE. La sección más crítica es la decorrelación, siendo la que más tiempo de ejecución requiere con la implementación actual. Se podría incluir la remoción de frecuencias intrínsecas tal como propone el equipo de BRITE y la decorrelación 2D.
- Aumentar el número de clases el dataset sintético incluyendo, por ejemplo, binarias eclipsantes o estrellas variables.
- Obtener o estimar los coeficientes de *limb darkening* de las estrellas seleccionadas a través de un modelo o catálogo. El procedimiento implementado utiliza un valor aleatorio dentro de unos rangos fijos.
- Identificar una restricción para permitir radios mayores y así poder comparar los resultados con los objetivos estudiados por [Yeh y Jiang, 2020].
- Generar tránsitos sintéticos para periodos más largos.
- Búsqueda exhaustiva en el resto de objetivos, por ejemplo, priorizando aquellos con menor dispersión de magnitudes o aquellos donde se generó un ejemplo PC²⁷.
- Aplicar métodos estadísticos sobre las señales encontradas, por ejemplo, un proceso de *sampling* Markov-Chain Monte-Carlo.

²⁷HD105211, HD106983, HD110879, HD111123, HD112092, HD114911, HD118716, HD128620, HD128621, HD128898, HD159492, HD160032, HD16970, HD172167, HD173648, HD175362, HD177196, HD178449, HD18331, HD192640, HD195068, HD198639, HD201092, HD201433, HD206267, HD20630, HD211336, HD214168, HD215664, HD217014, HD218396, HD22049, HD23850, HD27459, HD28052, HD32537, HD35296, HD35708, HD36861, HD39060, HD39587, HD4614, HD55892, HD61068, HD6961, HD74375, HD74956.

ANEXOS

6.1. ALGORITMOS DE TASAS ADAPTATIVAS

```
1
2 def AdaGrad(derivative_function, n_parameters, n_iterations, step_size,
3             initial_solution):
4     solution = initial_solution
5     r = init r as vector of zeros of lenght n_parameters
6
7     for _ in range(n_iterations): # do gradient descent
8
9         # calculate gradient
10        gradient = derivative_function(solution)
11
12        # update r with gradient magnitude information
13        # squared for each parameter
14        for j in range(n_parameters):
15            r[j] += gradient[j]**2
16
17        new_solution = init new solution with zero
18        for j in range(n_parameters): # update each parameter
19            new_solution[j] = solution[j] -
20                step_size / sqrt(r[j] + 1e-8)*gradient[j]
21
22        solution = new_solution
```

Listing 6.1: Pseudocódigo de algoritmo Adagrad.

```
1
2 def RMSProp(derivative_function, n_parameters, n_iterations, step_size,
3             initial_solution, rho):
4     solution = initial_solution
5     r = init r as vector of zeros of lenght n_parameters
6     for _ in range(n_iterations): # do gradient descent
7         # calculate gradient
8         gradient = derivative_function(solution)
9
10        # update r with moving average and gradient squared
11        for j in range(n_parameters):
12            r[j] = rho*r[j] + ( 1 - rho )*gradient[j]**2
13
14        new_solution = init new solution with zero
15        for j in range(n_parameters): # update each parameter
16            new_solution[j] = solution[j] -
17                step_size / sqrt(r[j] + 1e-8)*gradient[j]
18        solution = new_solution
```

Listing 6.2: Pseudocódigo de algoritmo RMSProp.

```

1
2 def Adam(objective, derivative, n_parametros, n_iterations, step_size,
  initial_solution, rho, beta, correction = False):
3
4     solution = initial_solution
5     r = init r as vector of zeros of lenght n_parameters
6     r_ = init r_ as vector of zeros of lenght n_parameters
7     v = init v as vector of zeros of lenght n_parameters
8     v_ = init v_ as vector of zeros of lenght n_parameters
9
10    for t in range(n_iterations): # do gradient descent
11
12        # calculate gradient
13        gradient = derivative_function(solution)
14
15        # update r with moving average and gradient squared,
16        # and speed with moving average and gradient
17
18        for j in range(n_parameters):
19            v[j] = beta*v[j] + (1-beta)*gradient[j]
20            r[j] = rho*r[j] + (1-rho)*gradient[j]**2
21
22        new_solution = init new solution with zero
23
24
25        if (correction):
26            # correct the bias of the variables by dividing
27            # by the constants
28            for i in range(n_parameters):
29                v_[j] = 1/(1-beta**(t+1)) * v[j]
30                r_[j] = 1/(1-rho**(t+1)) * r[j]
31
32            for j in range(n_parameters): # update each parameter
33                new_solution[j] = solution[j] -
34                    step_size / np.sqrt(r_[j] + 1e-8)*v_[j]
35        else:
36            for j in range(n_parameters): # update each parameter
37                new_solution[j] = solution[j] -
38                    step_size / np.sqrt(r[j] + 1e-8)*v[j]
39
40        solution = new_solution

```

Listing 6.3: Pseudocódigo de algoritmo Adam.

6.2. CASOS ESPECIALES EN BRITE

Algunos archivos públicos de BRITE requieren un tratamiento manual debido a 3 factores:

- Inconsistencia dentro del archivo.
- Calidad de los datos.
- Errores durante la ejecución del *pipeline* de preprocesamiento.

El detalle de estos casos se presenta a continuación:

- Los archivos del sector 16-Sgr-II-2016_DR5 estan vacios.
- HD108903_15-CruCar-I-2016_BAb_setupB_APa3s2chop_DR5.dat tiene un error en el header, se encuentra repetido. Se remueve este archivo.
- HD14386_21-CetEri-I-2016_BHr_setup1_APa3s2chop_DR5.dat y HD14386_21-CetEri-I-2016_BHr_setup2_APa3s2chop_DR5.dat no poseen valores de FLUX (tienen asteriscos).
- Los archivos HD93308_43-Car-III-2019_BTr_setup1_APa3s2chop_DR5.dat y HD93308_43-Car-III-2019_BTr_setup2_APa3s2chop_DR5.dat del sector 41-CMaPup-II-2018_DR5a estan vacios.
- Algunos archivos tienen el valor de ROlypos como "nm", se cambian a -1, por ejemplo, HD192910_04-Cyg-I-2014_BTr_setup1_APa2s5_DR2.dat y HD192910_04-Cyg-I-2014_UBr_setup1_APa2s5_DR2.dat
- Se remueve manualmente HD8538_11-CasCep-I-2015_BHr_setup1_APa3s2chop_DR4.dat porque tiene muy pocos datos (35 datos temporales).
- Durante el pipeline de preprocesamiento, los siguientes archivos gatillan un error: 11-CasCep-I-2015_DR4 HD205021_11-CasCep-I-2015_BHr_setup3_APa3s2chop_DR4.dat, 39-Cas-III-2018_DR5 HD205021_39-Cas-III-2018_BTr_setup2_APa3s2chop_DR5.dat, 29-LacCyg-I-2017_DR5 HD211336_29-LacCyg-I-2017_BTr_setup1_APa3s2chop_DR5.dat, 24-Car-I-2017_DR6 HD80007_24-Car-I-2017_BAb_setup3_APa6s1chop_DR6.dat, 24-Car-I-2017_DR6 HD80007_24-Car-I-2017_BAb_setup4_APa6s1chop_DR6.dat, 15-CruCar-I-2016_DR4 HD89080_15-CruCar-I-2016_BTr_setup2_APa3s2chop_DR4.dat, 62-CruCar-III-2021_DR5 HD96670_62-CruCar-III-2021_BHr_setup3_APa3s2chop_DR5.dat, 15-CruCar-I-2016_DR4 HD98718_15-CruCar-I-2016_BTr_setup2_APa3s2chop_DR4.dat Estos archivos son ignorados y no se utilizan.

6.3. RADIOS ESTELARES

Target	Radius [R_{sun}]	Target	Radius [R_{sun}]	Target	Radius [R_{sun}]
HD 017163	1.76896	HD 184006	2.84966	HD 192640	1.93986
HD 022468	3.03985	HD 159492	1.7583	HD 175813	1.92233
HD 017943	2.5702	HD 177196	1.8079	HD 020630	0.945495
HD 157919	3.01831	HD 181296	1.61681	HD 212075	2.91218
HD 056022	2.05634	HD 204188	1.45618	HD 027290	1.66959
HD 019107	1.88474	HD 218396	1.49338	HD 030739	2.82687
HD 024863	2.98215	HD 027459	2.06735	HD 028910	2.6756
HD 213617	1.6677	HD 027397	1.91138	HD 029388	2.70312
HD 040292	1.7289	HD 219134	0.750598	HD 178449	2.48067
HD 027628	1.81647	HD 032977	2.27987	HD 178253	2.10195
HD 201092	0.689846	HD 192696	2.69923	HD 012216	2.63923
HD 215664	2.41518	HD 032537	1.62058	HD 030478	2.97352
HD 201433	2.50732	HD 105211	1.78292	HD 054118	2.63153
HD 220974	2.17348	HD 201601	2.08786	HD 022634	2.77316
HD 218155	1.72835	HD 020320	1.99466	HD 027962	2.3842
HD 160032	1.62138	HD 205924	2.10283	HD 173648	3.02036
HD 035296	1.17966	HD 187013	1.5977	HD 000761	2.68915
HD 019319	2.56654	HD 006961	2.52427	HD 027934	2.95956
HD 025346	2.79004	HD 210074	2.66305	HD 206155	2.48905
HD 195068	1.63919	HD 028527	2.27881	HD 039060	1.54416
HD 206043	1.58642	HD 022049	0.755368	HD 033641	2.53095
HD 015633	2.47114	HD 031283	2.74672	HD 028052	3.06298
HD 055892	1.60696	HD 109787	2.67292	HD 211336	2.08817
HD 020888	1.61995	HD 018331	2.36828	HD 004614	1.10787
HD 198639	1.91513	HD 176723	2.77749	HD 162517	2.6806
HD 186155	3.01307	HD 032301	2.92729	HD 027463	2.7585
HD 038899	2.82167	HD 157792	1.91432	HD 210418	2.40784
HD 207223	1.81722	HD 217014	1.17561	HD 172167	2.45959

Tabla 6.1: Estrellas cuyo radio es menor o igual a 3.08 radio solar (Tabla 1/2). Fuente: Elaboración propia según lo descrito en el SubCapítulo 3.2.1.

Target	Radius [R_{sun}]	Target	Radius [R_{sun}]
HD 022484	1.84115	HD 045871	1.43
HD 039587	0.993029	HD 046328	2.27
HD 030780	2.63623	HD 061068	2.01
HD 202444	2.31533	HD 063922	2.36
HD 128621	1.22533	HD 074375	2.41
HD 020395	1.480739	HD 083446	2.1
HD 207652	1.782095	HD 088661	2.59
HD 000123	1.0	HD 105435	2.46
HD 000432	2.9	HD 106983	0.98
HD 012311	2.1	HD 110879	1.67
HD 016620	0.98	HD 111123	1.82
HD 016970	1.8	HD 112092	1.01
HD 023850	2.0	HD 118716	1.96
HD 035149	2.2	HD 122532	1.72
HD 035715	2.2	HD 157042	1.57
HD 074956	2.6	HD 168733	1.94
HD 114911	2.6	HD 173607	2.21
HD 128620	1.3	HD 175362	1.28
HD 128898	2.1	HD 190429	0.92
HD 155203	1.6	HD 192685	3.01
HD 198084	2.0	HD 206267	1.5
HD 203280	2.2	HD 209790	2.58
HD 214923	2.7	HD 214168	1.63
HD 022920	2.39	HD 216916	1.78
HD 029305	2.37	HD 225289	2.42
HD 035708	1.39	-	-
HD 036653	3.0	-	-
HD 036861	1.9	-	-

Tabla 6.2: Estrellas cuyo radio es menor o igual a 3.08 radio solar (Tabla 2/2). Fuente: Elaboración propia según lo descrito en el SubCapítulo 3.2.1.

REFERENCIAS BIBLIOGRÁFICAS

- [AAVSO, 2010] AAVSO (2010). Variable star astronomy. chapter 12: Variable stars and phase diagrams. <https://www.aavso.org/education/vsa>. [Online; accessed 12-jun-2022].
- [Aggarwal, 2018] Aggarwal, C. (2018). *Neural networks and deep learning*. Springer.
- [Aigrain y Favata, 2002] Aigrain, S. y Favata, F. (2002). Bayesian detection of planetary transits-a modified version of the gregory-loredo method for bayesian periodic signal detection. *Astronomy & Astrophysics*, 395(2):625–636.
- [Alen, 2021] Alen (2021). Guía básica de nanosatélites - alen. <https://alen.space/es/guia-basica-nanosatelites/>. [Online; accessed 20-nov-2021].
- [AstroLab, 2003] AstroLab, K. (2003). Guía básica de nanosatélites - alen. <https://www.astro.keele.ac.uk/astrolab/manual/manual.html>. [Online; accessed 15-may-2022].
- [Borucki, 2016] Borucki, W. J. (2016). Kepler mission: development and overview. *Reports on Progress in Physics*, 79(3):036901.
- [Bowman et al., 2022] Bowman, DM and Vandenbussche, B and Sana, H and Tkachenko, A and Raskin, G and Delabie, T and Vandoren, B and Royer, P and Garcia, S and Van Reeth, T and others (2022). The cubespec space mission: I. asteroseismology of massive stars from time-series optical spectroscopy: Science requirements and target list prioritisation. *Astronomy & Astrophysics*, 658:A96.
- [Christiansen, 2012] Christiansen, J. L. (2012). Kepler q1-q12 tce release notes.
- [Claret, 2003] Claret, A. (2003). A new non-linear limb-darkening law for lte stellar atmosphere models ii. *A&A*, 401:657–660.
- [Cryer y Chan, 2008] Cryer, J. D. y Chan, K.-S. (2008). *Time series analysis: with applications in R*, volumen 2. Springer.
- [D. Andrešič, 2021] D. Andrešič, P. Šaloun, B. S. (2021). Intelligent astrophysics (emergence, complexity and computation, 39).
- [de Données astronomiques de Strasbourg, 2022] de Données astronomiques de Strasbourg, C. (2022). Tess input catalog - v8.0 (tic-8) : lv/38. <https://cdsarc.cds.unistra.fr/viz-bin/cat/IV/38#/description>. [Online; accessed 01-nov-2022].
- [Douglas et al., 2019] Douglas, E. S., Cahoy, K. L., Knapp, M., y Morgan, R. E. (2019). Cubesats for astronomy and astrophysics. *arXiv preprint arXiv:1907.07634*.
- [Feigelson y Babu, 2012] Feigelson, E. D. y Babu, G. J. (2012). *Modern statistical methods for astronomy: with R applications*. Cambridge University Press.

- [Hippke *et al.*, 2019] Hippke, M., David, T. J., Mulders, G. D., y Heller, R. (2019). Wōtan: Comprehensive Time-series Detrending in Python. , 158(4):143.
- [Knapp *et al.*, 2020] Knapp, Mary and Seager, Sara and Demory, Brice-Olivier and Krishnamurthy, Akshata and Smith, Matthew W and Pong, Christopher M and Bailey, Vanessa P and Donner, Amanda and Di Pasquale, Peter and Campuzano, Brian and others (2020). Demonstrating high-precision photometry with a cubesat: Asteria observations of 55 cancri e. *The astronomical journal*, 160(1):23.
- [Krishnamurthy *et al.*, 2021] Krishnamurthy, Akshata and Knapp, Mary and Günther, Maximilian N and Daylan, Tansu and Demory, Brice-Olivier and Seager, Sara and Bailey, Vanessa P and Smith, Matthew W and Pong, Christopher M and Hughes, Kyle and others (2021). Transit search for exoplanets around alpha centauri a and b with asteria. *The astronomical journal*, 161(6):275.
- [Kunimoto *et al.*, 2022] Kunimoto, M., Winn, J., Ricker, G. R., y Vanderspek, R. K. (2022). Predicting the exoplanet yield of the tess prime and extended missions through years 1–7. *The Astronomical Journal*, 163(6):290.
- [Lang, 2013] Lang, K. R. (2013). *Essential astrophysics (Undergraduate Lecture Notes in Physics)*. Springer.
- [Mandel y Agol, 2002] Mandel, K. y Agol, E. (2002). Analytic light curves for planetary transit searches. *The Astrophysical Journal*, 580(2):L171.
- [Mitchell, 1997] Mitchell, T. M. (1997). *Machine learning*. McGraw-hill New York.
- [Murphy, 2012] Murphy, S. J. (2012). An examination of some characteristics of kepler short- and long-cadence data. *Monthly Notices of the Royal Astronomical Society*, 422(1):665–671.
- [NASA, 2023] NASA (2023). Nasa exoplanet exploration. <https://exoplanets.nasa.gov/>. [Online; accessed 17-jul-2023].
- [Pablo *et al.*, 2016] Pablo, H and Whittaker, GN and Popowicz, A and Mochnacki, SM and Kuschnig, Rainer and Grant, CC and Moffat, AFJ and Rucinski, SM and Matthews, JM and Schwarzenberg-Czerny, A and others (2016). The brite constellation nanosatellite mission: testing, commissioning, and operations. *Publications of the Astronomical Society of the Pacific*, 128(970):125001.
- [Pigulski, 2018] Pigulski, A. (2018). Brite cookbook 2.0. *arXiv preprint arXiv:1801.08496*.
- [Popowicz *et al.*, 2017] Popowicz, Adam and Pigulski, Andrzej and Bernacki, Krzysztof and Kuschnig, Rainer and Pablo, Herbert and Ramiamananantsoa, Tahina and Zocłńska, E and Baade, Dietrich and Handler, Gerald and Moffat, AFJ and others (2017). Brite constellation: data processing and photometry. *Astronomy & Astrophysics*, 605:A26.

- [Ricker *et al.*, 2016] Ricker, George R and Vanderspek, R and Winn, J and Seager, S and Bert-Thompson, Z and Levine, A and Villaseñor, J and Latham, D and Charbonneau, D and Holman, M and others (2016). The transiting exoplanet survey satellite. En *Space Telescopes and Instrumentation 2016: Optical, Infrared, and Millimeter Wave*, volumen 9904, pp. 767–784. SPIE.
- [Serjeant *et al.*, 2020] Serjeant, S., Elvis, M., y Tinetti, G. (2020). The future of astronomy with small satellites. *Nature Astronomy*, 4(11):1031–1038.
- [Shallue y Vanderburg, 2018] Shallue, C. J. y Vanderburg, A. (2018). Identifying exoplanets with deep learning: A five-planet resonant chain around kepler-80 and an eighth planet around kepler-90. *The Astronomical Journal*, 155(2):94.
- [Shkolnik, 2018] Shkolnik, E. L. (2018). On the verge of an astronomy cubesat revolution. *Nature Astronomy*, 2(5):374–378.
- [Sódor y Bognár, 2020] Sódor, Á. y Bognár, Z. (2020). The planet-host pulsating star hr 8799 as seen by brite. *Stars and their Variability Observed from Space*, pp. 91–92.
- [Sódor *et al.*, 2014] Sódor, Ádám and Chené, A-N and De Cat, P and Bognár, Zs and Wright, DJ and Marois, C and Walker, GAH and Matthews, JM and Kallinger, T and Rowe, JF and others (2014). Most light-curve analysis of the γ doradus pulsator hr 8799, showing resonances and amplitude variations. *Astronomy & Astrophysics*, 568:A106.
- [Stassun *et al.*, 2018] Stassun, Keivan G and Oelkers, Ryan J and Pepper, Joshua and Paegert, Martin and De Lee, Nathan and Torres, Guillermo and Latham, David W and Charpinet, Stéphane and Dressing, Courtney D and Huber, Daniel and others (2018). The tess input catalog and candidate target list. *The Astronomical Journal*, 156(3):102.
- [Vanderburg y Johnson, 2014] Vanderburg, A. y Johnson, J. A. (2014). A technique for extracting highly precise photometry for the two-wheeled kepler mission. *Publications of the Astronomical Society of the Pacific*, 126(944):948.
- [Weiss *et al.*, 2014] Weiss, Werner W and Rucinski, SM and Moffat, AFJ and Schwarzenberg-Czerny, A and Koudelka, OF and Grant, CC and Zee, RE and Kuschnig, Rainer and Matthews, JM and Orleanski, P and others (2014). Brite-constellation: nanosatellites for precision photometry of bright stars. *Publications of the Astronomical Society of the Pacific*, 126(940):573.
- [Wright y Gaudi, 2012] Wright, J. T. y Gaudi, B. S. (2012). Exoplanet detection methods. *arXiv preprint arXiv:1210.2471*.
- [Yeh y Jiang, 2020] Yeh, L.-C. y Jiang, G. (2020). Searching for possible exoplanet transits from brite data through a machine learning technique. *Publications of the Astronomical Society of the Pacific*, 133(1019):014401.

[Yu *et al.*, 2019] Yu, Liang and Vanderburg, Andrew and Huang, Chelsea and Shallue, Christopher J and Crossfield, Ian JM and Gaudi, B Scott and Daylan, Tansu and Dattilo, Anne and Armstrong, David J and Ricker, George R and others (2019). Identifying exoplanets with deep learning. iii. automated triage and vetting of tess candidates. *The Astronomical Journal*, 158(1):25.