UNIVERSITAT POLITÈCNICA DE VALÈNCIA

MIARFID Máster en Inteligencia Artificial, Reconocimiento de Formas e Imagen Digital

Departamento de Sistemas Informáticos y Computación
Universidad Politécnica de Valencia

Búsqueda y detección de exoplanetas mediante técnicas de Machine Learning

TRABAJO FIN DE MASTER

Master Universitario en Inteligencia Artifical, Reconocimiento de Formas e Imagen Digital

Autor: Raúl García Crespo

Tutores: Roberto Paredes Palacios
Roberto Ruiz de Austri Bazan

Curso 2020-2021

Resumen

El termino 'exoplaneta' se emplea para definir todos aquellos planetas que orbiten una estrella distinta al Sol. El descubrimiento de estos planetas ha fomentado el interés en la búsqueda de vida alienígena, pues analizando la composición de estos planetas y teniendo en cuenta factores como la distancia entre este y su estrella, es posible deducir teóricamente si podrían contener vida. Por desgracia, la detección de estos planetas solamente puede llevarse a cabo mediante estrictas observaciones de las estrellas que orbitan. Por ejemplo, observando los descensos periódicos en la luz emitida por la estrella, provocados por el paso del exoplaneta frente a ella.

Este trabajo de fin de máster se centra en el estudio y el desarrollo de distintas herramientas, basadas en redes neuronales artificiales, que sean capaces de diferenciar las variaciones de radiación emitida por una estrella causadas por el desplazamiento de un exoplaneta, de aquellas variaciones habituales y presentes en todas las estrellas. Con este objetivo, hemos desarrollado un modelo capaz de confirmar o descartar la presencia de exoplanetas en otros sistemas solares. Para ello, hacemos uso de un conjunto de observaciones de la radiación emitida por cientos de estrellas, las cuales han sido etiquetadas de antemano, indicando la presencia o no de un exoplaneta en ellas. Este etiquetado nos ha permitido realizar un entrenamiento supervisado de nuestros modelos y ha facilitado la validación de los mismos.

Adicionalmente, ha sido necesario resolver un problema característico de la búsqueda de exoplanetas: la disparidad en las observaciones disponibles. Dado que la mayoría de las estrellas observables no poseen planetas, solo se dispone de un conjunto relativamente pequeño de observaciones de exoplanetas. Otros trabajos de carácter similar al nuestro resuelven este inconveniente mediante el uso de datos simulados, los cuales permiten aumentar artificialmente el número de observaciones disponibles. Sin embargo, nosotros creemos que este tipo de práctica puede perjudicar los resultados de los modelos, por lo que hemos optado por invertir esfuerzo en el desarrollo de un proceso de tratamiento de los datos que nos permita emplear muestras reales.

Palabras clave: Machine Learning; Red Neuronal; Exoplaneta; Astrofísica; Visión por Computador

Resum

El terme 'exoplaneta' s'empra per a definir tots aquells planetes que orbiten una estrela diferent al Sol. El descobriment d'aquests planetes ha fomentat l'interés en la cerca de vida alienígena, perquè analitzant la composició d'aquests planetes i tenint en compte factors com la distància entre aquest i la seua estrela, és possible deduir teòricament si podrien contindre vida. Per desgràcia, la detecció d'aquests planetes solament pot dur-se a terme mitjançant estrictes observacions de les estreles que orbiten. Per exemple, observant els descensos periòdics en la llum emesa per l'estrela, provocats pel pas de l'exoplaneta enfront d'ella.

Aquest treball de fi de màster se centra en l'estudi i el desenvolupament de diferents eines, basades en xarxes neuronals artificials, que siguen capaces de diferenciar les variacions de radiació emesa per una estrela causades pel desplaçament d'un exoplaneta, d'aquelles variacions habituals i presents en totes les estreles. Amb aquest objectiu, hem desenvolupat un model capaç de confirmar o descartar la presència d'exoplanetes en altres sistemes solars. Per a això, fem ús d'un conjunt d'observacions de la radiació emesa per centenars d'estreles, les quals han sigut etiquetats per endavant indicant la presència o no d'un exoplaneta en elles. Aquest etiquetatge ens ha permés realitzar un entrenament supervisat dels nostres models i ha facilitat la validació d'aquests.

Addicionalment, ha sigut necessari resoldre un problema característic de la cerca d'exoplanetes: la disparitat en les observacions disponibles. Atés que la majoria de les estreles observades no posseeixen planetes, només es disposa d'un conjunt relativament reduït d'observacions d'exoplanetes. Altres treballs de caràcter similar al nostre resolen aquest inconvenient mitjançant l'ús de dades simulades, els quals permeten augmentar artificialment el nombre d'observacions disponibles. No obstant això, nosaltres creiem que aquest tipus de pràctica pot perjudicar els resultats dels models, per la qual cosa hem optat per invertir esforç en el desenvolupament d'un procés de tractament de les dades que ens permeta emprar mostres reals.

Paraules clau: Machine Learning; Xarxa Neuronal; Exoplaneta; Astrofísica; Visió per Computador

Abstract

The term 'exoplanet' is used to define all those planets that orbit a star other than the Sun. The discovery of these planets has fostered interest in the search for alien life, given that by analyzing the composition of these planets and taking into account factors such as the distance between them and its star, it is possible to deduce theoretically if they could host life. Unfortunately, detection of these planets can only be accomplished by strict observations of the stars they orbit. For example, observing the periodic decreases in the light emitted by the star, caused by the passage of the exoplanet in front of it.

This master's thesis focuses on the study and development of different tools, based on artificial neural networks, that are capable of differentiating the variations in radiation emitted by a star caused by the displacement of a planet, from those habitual and present in all the stars. With this objective, we have developed a model capable of confirming or ruling out the presence of exoplanets in other solar systems. To do this, we make use of a set of observations of the radiation emitted by hundreds of stars, which will have been labeled in advance indicating the presence or not of an exoplanet in them. This labeling has allowed us to carry out a supervised training of our models and has facilitated their validation.

Additionally, it has been necessary to solve a characteristic problem of the search for exoplanets: the disparity in the available observations. Since most of the observed stars do not have planets, only a relatively small set of exoplanet observations is available. Other works of a similar nature to ours solve this problem by using simulated data, which allow the number of available observations to be artificially increased. However, we believe that this type of practice can harm the results of the models, so we have chosen to invest effort in developing a data treatment process that allows us to use real samples.

Key words: Machine Learning; Neural Network; Exoplanet; Astrophysics; Computer Vision

Índice general

Índice de figuras

Índice de tablas

CAPÍTULO 1
Introducción

Desde la primera confirmación de un planeta ajeno al Sistema Solar en 1992 [1], se han detectado con exito más de 4000 exoplanetas[1] aunando los esfuerzos de distintas misiones espaciales y el trabajo de distintos grupos de investigación. Estos planetas que orbitan estrellas a distancias de decenas, cientos o miles de años luz de nosotros se encuentran demasiado alejados para poder ser observados de forma directa. La detección de exoplanetas requiere analizar en detalle ciertas características de la estrella a la que orbitan para poder, de forma indirecta, detectar patrones o factores indicativos de la presencia de estos.

Los métodos de detección de exoplanetas más habituales hasta la fecha son los métodos basados en velocidad radial y tránsito astronómico. El primero de estos métodos se centra en observar cambios en la velocidad de una estrella causados por los efectos gravitatorios de un posible planeta que la orbite. Por su parte, el segundo método obtiene sus resultados a partir de las variaciones de radiación que somos capaces de percibir desde la estrella, generadas por el exoplaneta durante su tránsito entre la estrella y nuestro punto de vista. Estos dos no son los únicos métodos, pero si los más empleados [2], siendo la detección por velocidad radial el método más empleado en el pasado, mientras que la exploración mediante tránsitos astronómicos ha ido ganando importancia hasta convertirse en prácticamente el método estándar en la actualidad.

La aplicación de estos métodos requiere recopilar previamente datos e información de estrellas lejanas durante largos periodos de tiempo, por lo que suele ser común programar misiones que estudien durante años un mismo grupo de estrellas, a fin de obtener esta información. Proyectos como estos corresponden por ejemplo a las misiones Kepler, K2[2] o TESS[3], las cuales observaron durante años las estrellas presentes en secciones concretas del cielo nocturno con el propósito de medir la variación de distintos factores de estas a lo largo del tiempo.

Por lo tanto, solo un número relativamente pequeño de estrellas puede ser estudiado en la búsqueda de exoplanetas, y ese proceso debe durar al menos varios años hasta disponer de los datos necesarios para poder comenzar a estudiarlos con detalle. Como complejidad adicional, confirmar la presencia de un exoplaneta por métodos externos a los mismos que se han empleado para proponer su presencia es altamente complicado, pues no suele ser posible observar visualmente dicho planeta. La confirmación habitualmente requiere que varios grupos de investigación analicen los datos disponibles y alcancen las mismas conclusiones en investigaciones paralelas, pero independientes, antes de aceptar la presencia de uno de estos planetas.

[1] https://exoplanets.nasa.gov/faq/6/how-many-exoplanets-are-there/
[2] https://www.nasa.gov/mission_pages/kepler/overview/index.html
[3] https://tess.mit.edu/

En la actualidad, los avances en el campo de las redes neuronales nos permiten desarrollar modelos orientados a la clasificación de datos, definidos específicamente en función de sus características. Para este trabajo, en el que se han empleado como datos de partida las series temporales correspondientes a las variaciones del flujo detectado desde distintas estrellas, disponemos de diferentes modelos capaces de trabajar con este tipo de datos y que han demostrado obtener buenos resultados en tareas similares. Si bien anteriormente el estado del arte en el campo de la clasificación y predicción de series temporales pasaba por el empleo de las Redes Recurrentes y los modelos con capas tipo LSTM, en los últimos años se ha demostrado que las redes basadas en convoluciones 1D pueden alcanzar incluso mejores resultados en estas mismas tareas [20].

Empleando como *input* de estos modelos los mismos datos empleados en los métodos de detección clásicos, podría ser posible usar estos como herramienta de cribado, que ayuden a los equipos de investigación a filtrar aquellas estrellas con menos probabilidad de poseer un exoplaneta. Dependiendo de su precisión, incluso podría automatizarse la detección de estos astros.

1.1 Motivación

Este trabajo persigue aplicar algunos de los más recientes avances en inteligencia artificial y técnicas de *Machine Learning (ML)* a la búsqueda de exoplanetas, empleando para ello datos reales obtenidos mediante el uso de potentes telescopios espaciales, detectados en diferentes misiones realizadas por la NASA, y accesibles al dominio público. Con ello, se espera estudiar la eficacia de estos métodos frente a los empleados en la actualidad, basados en costosos *pipelines* que requieren un estudio detallado de las señales recibidas desde otras estrellas, y encontrar aquel método que maximice la precisión a la hora de confirmar la existencia de un exoplaneta.

Se espera que los resultados y conclusiones aquí alcanzados puedan servir como indicadores de la utilidad de incluir distintas técnicas de *Machine Learning* en los métodos clásicos de detección de exoplanetas, así como desarrollar modelos clasificadores que puedan convertirse en herramientas útiles a la hora de analizar la información de las cientos de miles de estrellas disponibles en este momento.

1.2 Objetivos

Los objetivos que se plantean alcanzar durante la realización de este trabajo son:

- Explorar los archivos públicos actuales, seleccionar aquellos datos con mayor potencial y preparar estos para que sean adecuados en nuestro estudio. En conjunto, este objetivo requiere generar un *data set* de entrenamiento y test con muestras correctamente etiquetadas que actuarán como *input* de nuestros modelos.
- Analizar estudios similares con el propósito de obtener un punto de partida para la construcción de nuestros modelos clasificadores.
- Construir y evaluar distintos modelos clasificadores que, a partir de datos concretos de una estrella, sean capaces de clasificar correctamente la existencia o no de un exoplaneta alrededor de dicha estrella.
- En base al estado del arte, se compararán los resultados obtenidos con los de otros trabajos similares.

1.3 Estructura

La estructura organizativa detrás de esta memoria de Trabajo de Final de Máster comienza con una descripción del estado del arte en el Capítulo 2, donde se describen y enumeran trabajos de carácter similar al nuestro, centrados en la exploración astrofísica desde el punto de vista del *Machine Learning*. Concretamente, en este primer capítulo se listan las principales técnicas y modelos más comúnmente aplicados a la detección de exoplanetas, al mismo tiempo que se dedica un breve espacio a introducir y describir las arquitecturas neuronales que finalmente hemos empleado para obtener nuestro propios resultados.

La memoria continua con el Capítulo 3, donde se describe el proceso seguido para la obtención de los datos con los que hemos construido nuestro *data set* final. Se han empleado datos de distintas fuentes, cada una de las cuales recibe su propia sección dentro del capítulo, así como se incluye la relevancia de cualquier software o código externo del que hayamos hecho uso. Continuando a la descripción de las fuentes de datos, el Capítulo 4 detalla el tratamiento que los datos descargados han recibido, como se han filtrado y que medidas hemos tomado para asegurar la fiabilidad del entrenamiento y los resultados de los modelos.

Los modelos previamente introducidos en el Capítulo 2 son desarrollados en profundidad en el Capítulo 5, en el cual se definen los detalles y parámetros de las redes neuronales empleados durante la experimentación junto con la definición de las métricas con las que hemos comparado los resultados. Continuamos con el Capítulo 6 y la presentación de los resultados siguiendo las métricas descritas en el capítulo anterior, incluyendo varios comentarios sobre patrones y comportamientos que se han detectado en ellos.

Por último, este trabajo finaliza con el Capítulo 7, el cual detalla los conocimientos adquiridos durante la elaboración de este estudio, al mismo tiempo que evalúa si realmente se han alcanzado los objetivos que se habían planteado para él; y el Capítulo 8, en el que se ofrecen propuestas de ampliación en la forma de nuevas metodologías de experimentación que creemos que podrían aportar valor adicional a los resultados que hemos obtenido.

CAPÍTULO 2
Estado del arte

En esta sección del trabajo se exponen brevemente las técnicas más empleadas en el ámbito de la astrofísica aplicada a la detección de exoplanetas, y una exploración en mayor profundidad de las aplicaciones actuales del *Machine Learning* a este campo. Adicionalmente, se habla de las bases y estado actual de las tecnologías de modelos neuronales que finalmente se han empleado durante el desarrollo de este trabajo.

2.1 Detección de exoplanetas

Debido a la distancia que nos separa de ellos, los exoplanetas no suelen poder ser observados directamente, ni siquiera por los telescopios más potentes. En lugar de ello, es posible detectar su presencia observando el efecto indirecto que producen sobre la estrella que orbitan, la cual si es fácilmente observable a través de multitud de canales.

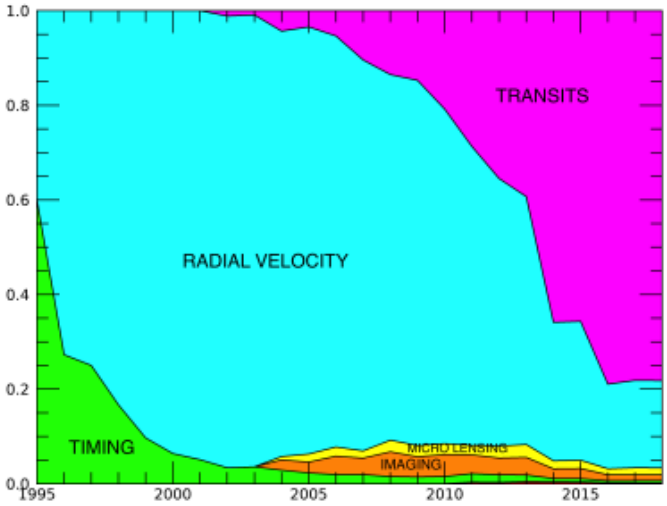

Figura 2.1: Esta gráfica muestra el porcentaje acumulado de uso de cada tipo de técnica empleada en la detección de exoplanetas entre 1995 y 2018 [3]

Desde el comienzo del estudio de estos astros, en los primeros años de la década de los 90, los procedimientos a seguir en el ámbito de la detección de exoplanetas han ido evolucionando. Si bien en los primeros trabajos publicados la mayoría de resultados se obtuvieron empleando métodos *Timing* y de velocidad radial, estos métodos presentan ciertos inconvenientes que limitan el alcance de su aplicación. Los métodos basados en *Timing* requieren que el sistema solar observado posea ciertas características especiales, tales como emitir ondas de radio continuamente o que se encuentre compuesto por una

pareja de estrellas (fenómeno conocido como "sistema binario"). Por otra parte, los sistemas en los que se han detectado exoplanetas empleando el método de velocidad radial requieren que la estrella observada posea un tamaño reducido en comparación al planeta, que debe poseer una gran masa, ya que son los efectos gravitatorios del planeta sobre la estrella lo que este método pretende detectar.

Estas limitaciones y los avances alcanzados en la construcción de telescopios espaciales propiciaron la aparición de un nuevo método, la detección mediante tránsitos astronómicos. Si bien esta técnica no esta libre de inconvenientes y limitaciones, permite detectar un mayor abanico de planetas, al mismo tiempo que facilita observar simultáneamente múltiples estrellas.

Tal y como muestra la figura 2.1 la técnica claramente predominante en la actualidad es la detección mediante tránsitos, siendo la empleada en más del 78 % de planetas detectados en el año 2018[3]. Es por este motivo que las bases desde las que partiremos en este trabajo serán similares a las que se emplean con esta técnica.

2.1.1. Técnica de tránsito astronómico

Los tránsitos astronómicos o planetarios hacen referencia al efecto observable durante el periodo en el que un cuerpo se interpone entre un astro y el observador, siendo el tipo de tránsito más conocido el eclipse solar. Un efecto similar a los eclipses puede observarse en estrellas más lejanas en el momento en el que uno de sus planetas atraviesa la linea de visión entre nuestros telescopios y la estrella. Si observamos continuamente la luz emitida por la estrella podemos detectar descensos periódicos durante estos periodos de tránsito, confirmando así de forma indirecta la presencia de un planeta.

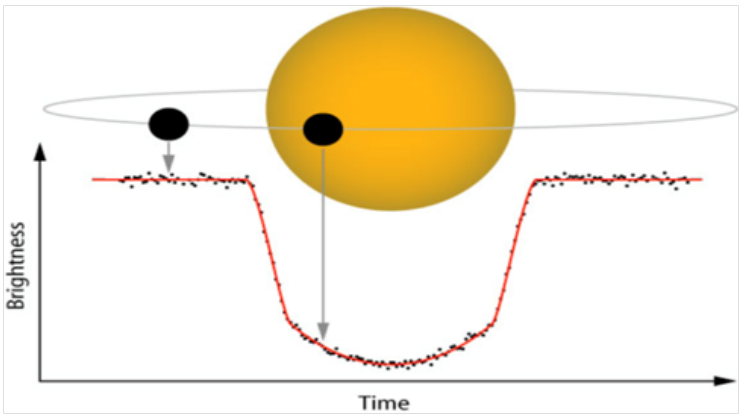

Figura 2.2: Representación de un periodo de tránsito planetario. Puede observarse como la cantidad de luz que percibe el observador desde la estrella desciende a medida que el planeta cruza entre ambos.

Para poder afirmar con exactitud que los efectos de un tránsito han sido generados por un planeta es necesario seguir observando la estrella a la espera de que estos fenómenos se produzcan de forma periódica, pues la duración del recorrido del planeta al orbitar la estrella debe ser constante. Esto condiciona que sea necesario observar la estrella a la espera de fenómenos de tránsito durante largos periodos de tiempo hasta obtener pruebas suficientes de la existencia del exoplaneta.

Con esta técnica no solo es posible demostrar la presencia de planetas, si no que también nos ofrece la posibilidad de recopilar información sobre el planeta en sí. Si calculamos la bajada en el flujo de luz que recibimos desde la estrella durante los periodos de tránsito podemos calcular el tamaño del planeta, y si realizamos un análisis espec-

trográfico de la luz recibida de la estrella durante ese mismo periodo podemos obtener información sobre la atmósfera del planeta, aunque estas cuestiones quedan fuera del ámbito de este trabajo.

Para poder aplicar esta técnica es necesario disponer de una secuencia temporal de valores de flujo correspondientes a una estrella, la cual se analizará en busca de periodos de tránsito. Esta secuencia, que recibe el nombre de grafo de flujo, será la que empleemos como entrada de los modelos neuronales que se han desarrollado a lo largo de este trabajo.

2.2 Aplicaciones actuales de *Machine Learning* a la detección de exoplanetas

Como exploración previa al desarrollo de este trabajo se han investigado las aplicaciones actuales del ML al ámbito de la búsqueda de exoplanetas.

2.2.1. Procesamiento inicial de la señal

Los datos de los que partiremos serán los mismos que han empleado otros investigadores en estudios similares: series temporales mostrando la variación en el flujo de luz observado en distintas estrellas. La observación de estos flujos no puede realizarse en la superficie terrestre, pues la atmósfera y la contaminación lumínica harían imposible detectar unos cambios tan sensibles. En lugar de ello, la vigilancia de las estrellas estudiadas se lleva a cabo a través de potentes telescopios espaciales, situados en orbita alrededor de la Tierra. Obviamente, el despliegue y uso de estos artefactos no está al alcance de cualquier institución; es por ello que, replicando la metodología observada en trabajos similares, vamos a trabajar con datos obtenidos por los telescopios espaciales lanzados y mantenidos por la NASA, los cuales son públicamente accesibles y han sido revisados exhaustivamente por otros investigadores.

Las tres misiones espaciales de la NASA dedicadas a la exploración de exoplanetas hasta la fecha, que reciben los nombres de *Kepler, K2* y *TESS*, han permitido mantener una exploración constante desde el año 2009 hasta la actualidad, con una pausa de un año en 2013 debido a dificultades técnicas en uno de los satélites/telescopios. Estos telescopios son capaces de observar grandes secciones del cielo nocturno simultáneamente, obteniendo información de más de 200.000 estrellas a intervalos periódicos.

A pesar de que la información registrada por estas misiones es pública, no la encontramos en el mismo estado en el que fue registrada por los instrumentos a bordo de los satélites. La NASA aplica un *pipeline* específico a cada uno de estos instrumentos, con el propósito de limpiar las señales observadas y eliminar fuentes de ruido conocidas, tales como las generadas por la propia maquinaria del telescopio. Estos *pipelines* aprovechan los datos en limpio para aplicar una búsqueda pre-eliminar de fenómenos de tránsito en los datos de cada estrella. Por ejemplo, en el informe de objetivos del software empleado en el *pipeline* de la misión *Kepler*[13], se detalla que tras la eliminación de errores sistemáticos en los datos se inicia un proceso sobre estos de búsqueda de fenómenos de tránsito planetarios. Según el informe, para esta búsqueda se emplea un algoritmo de reconocimiento de patrones basado en una transformada de ondícula, un tipo especial de transformada matemática que representa una señal en términos de versiones trasladadas y dilatadas de una onda finita. El resultado de este reconocimiento de patrones no es suficiente para confirmar la presencia de un exoplaneta en los datos, pero sí para construir

una lista de posibles estrellas candidatas [1] [2], la cual también es accesible al público y es habitual que sea el punto de partida de varios estudios para seleccionar los datos en los que centrar el foco de sus esfuerzos.

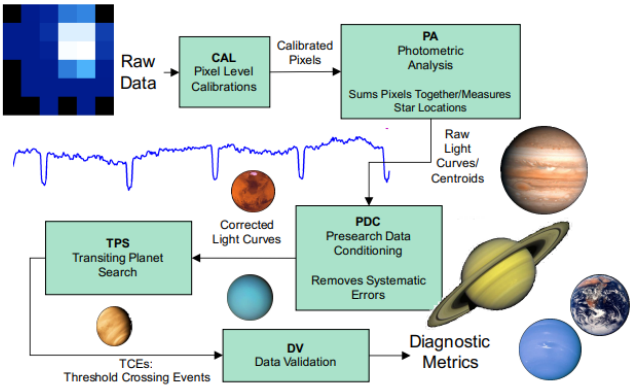

Figura 2.3: Esquema simplificado del *pipeline* de la misión *Kepler* tal y como aparece en el informe de objetivos [13]

2.2.2. Clasificación de muestras simuladas

Un inconveniente importante a la hora de construir modelos clasificadores capaces de diferenciar los patrones de un exoplaneta es la falta de muestras etiquetadas disponibles para entrenar dicho modelo. La búsqueda de planetas fuera del Sistema Solar es un proyecto relativamente reciente, activo desde principios de la década de los 90, pero de forma aislada en sus primeros años. Es en la década de los años 2000 cuando realmente comienzan a plantearse misiones y proyectos dedicados expresamente a esta tarea. Por ejemplo, la misión Kepler, la más exitosa hasta la fecha, llegó a registrar datos de más de 200.000 estrellas, entre las cuales se hallaron solamente en torno a 2300 exoplanetas. Si el objetivo de nuestro proyecto es resolver un problema de clasificación binaria a partir únicamente de los datos recibidos de cada estrella, nos encontramos con una enorme disparidad en el número de muestras disponibles de cada clase: tenemos acceso a más muestras negativas (sin presencia de exoplaneta) de las que necesitamos pero no a suficientes muestras positivas (con presencia de exoplaneta).

Para solventar este problema numerosos grupos de investigación han optado por construir su propio *data set* empleando las muestras negativas recogidas por los telescopios espaciales, pero simulando de forma artificial el conjunto de muestras positivas. Es posible seleccionar parte de las muestras que ya han sido confirmadas como negativas e insertar de forma artificial y aleatoria periodos de tránsito que simulen las bajadas en la radiación emitida por las estrellas. Con este método es posible lograr un *data set* de tamaño adecuado para esta tarea con una cantidad igualada de muestras en cada clase.

Los artículos publicados por Malik et al.[5] o Jiang et al.[6] emplean esta metodología, sin embargo el primero emplea técnicas más clásicas de ML, mientras que el segundo ataca el problema empleando modelos neuronales convolucionales. El trabajo de Malik et al. trabaja con un modelo clasificador basado en *Gradient Boosting Trees*, una técnica de aprendizaje automático que genera un modelo clasificador compuesto a partir de modelos más débiles, en este caso árboles de decisión. Este modelo permitió alcanzar unos resultados de precisión del 92 % para la clasificación de los datos simulados.

[1] https://exoplanetarchive.ipac.caltech.edu/cgi-bin/TblView/nph-tblView?app=ExoTbls&config=k2candidates
[2] https://exoplanetarchive.ipac.caltech.edu/cgi-bin/TblView/nph-tblView?app=ExoTbls&config=TOI

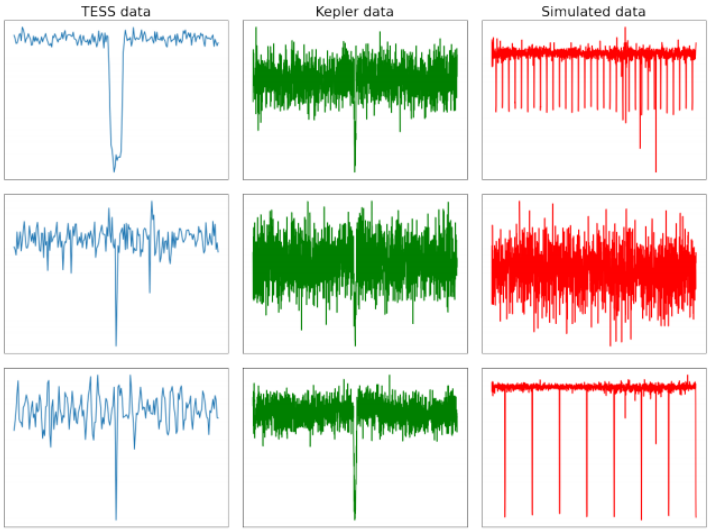

Figura 2.4: Comparación de secuencias de flujo simuladas frente a secuencias reales de las misiones Kepler y TESS presentadas en el trabajo *Exoplanet Detection using Machine Learning* de Malik et al.

Por su parte, el trabajo de Jiang et al. centra su esfuerzos en desarrollar modelos basados en redes neuronales convolucionales [7]. Este tipo de redes emplean principalmente tres tipos de capas:

- Capas convolucionales. Este tipo de capas aplican la multiplicación de un filtro/-kernel a distintas ventanas de los datos de entrada hasta extraer un mapa de características. Acciones opcionales, como añadir un margen a la entrada, el tamaño del kernel a emplear o la distancia sobre los datos de entrada entre las ventanas a las que se aplica el filtro, pueden provocar que la salida de la capa convolucional sea o no del mismo tamaño que la entrada.

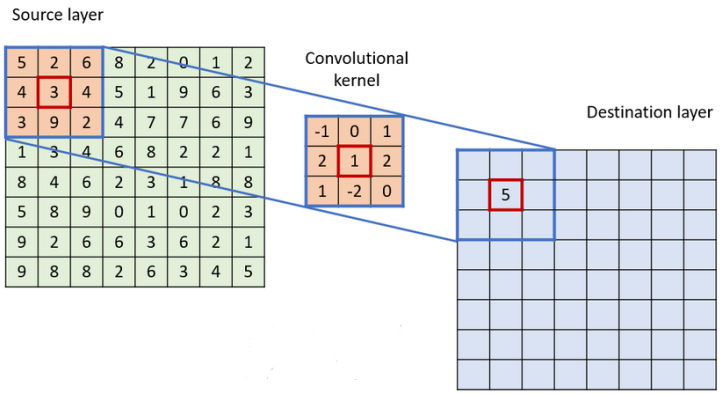

Figura 2.5: Diagrama de funcionamiento de una capa convolucional con una entrada en 2 dimensiones y un kernel de tamaño 3x3

- Capas *Pooling*. Estas capas reducen el tamaño de la entrada original aplicando distintas funciones. Las mayormente empleadas son el calculo del valor máximo (Capa Maxpooling) o el calculo de la media (Capa AveragePooling). Estas capas, al igual que las convolucionales, aplican su función sobre una ventana que se va desplazando sobre los datos de entrada, generando así la salida.

- Capas densas o *Fully Connected*. Cada una de las neuronas de esta capa se encuentran conectadas a cada uno de los valores que recibe como entrada. Cada enlace entre un valor de entrada y una neurona tiene asociado un peso que multiplica al valor original y genera el de salida. El valor de las neuronas de salida es la suma de los valores calculados por los enlaces que alcancen a dichas neuronas. Habitualmente se incluye una, función de activación a la salida de estas capas, limitando el rango de valores que pueden presentar como resultado.

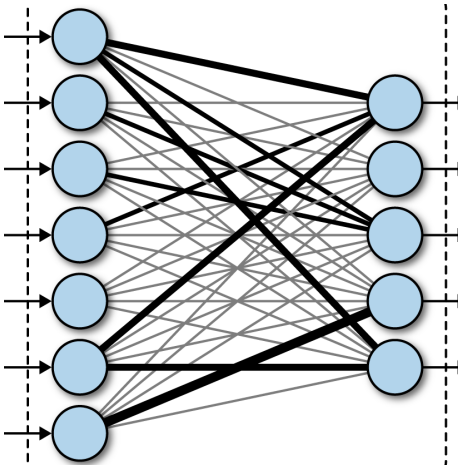

Figura 2.6: representación de las conexiones de una capa densa con 7 valores de entrada y 5 neuronas en la salida

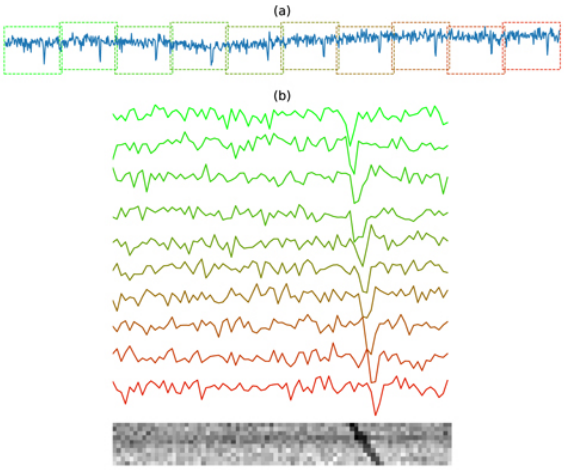

Figura 2.7: Segmentación de una señal undimensional para construir una imagen 2D. a) Secuencia original. b) Secuencia segmentada y representación en forma de matriz bidimensional

Dado que los datos de entrada al modelo en cada paso del entrenamiento son secuencias de valores de flujo, estas se representan como vectores unidimensionales, lo que permite que el primer modelo propuesto por estos autores haga uso de capas convolucionales 1D, también denominadas convoluciones temporales. Sin embargo, para explorar otras alternativas y adaptar los datos para emplear convoluciones 2D, los autores proponen segmentar las secuencias originales en fragmentos de igual longitud y recomponer estos en forma de imagen 2D como se muestra en la figura 2.7. Al emplear esta representación los autores proponen que la red debería de ser capaz de identificar patrones como los que observamos en la parte b) de la figura2.7, donde al apilar los segmentos, los periodos de tránsito generan una linea diagonal en la matriz. Esta aplicación poco ortodoxa de las redes convolucionales y la transformación propuesta de los datos puede ser útil

en este caso, ya que al emplear muestra simuladas es posible controlar la distancia entre periodos de tránsito, asegurándose de que estos cumplan las condiciones necesarias para repetirse al menos una vez por cada segmento de los datos.

2.2.3. Clasificación de muestras reales

Partiendo de los datos procesados, como se describe en el apartado 2.2.1, se han realizado varios estudios analizando la aplicabilidad de diferentes modelos neuronales a la detección de patrones propios de exoplanetas en los grafos de flujo, con diferentes enfoques y arquitecturas. Uno de dichos trabajos lo encontramos en la publicación de Shallue y Vanderburg [4], quienes parten con el objetivo de desarrollar un modelo capaz de diferenciar entre verdaderas detecciones de exoplanetas de aquellas asociadas a falsos positivos u otros eventos astronómicos, empleando como datos iniciales las observaciones correspondientes a las estrellas propuestas como candidatas por los distintos *pipelines* de la NASA. Al hacer uso solamente de las muestras que ya han sido propuestas como candidatas, los autores no emplean la totalidad de las muestras recogidas por las distintas misiones espaciales, únicamente aquellas con posibilidades de contener a exoplanetas. Esta limitación en los datos implica que los resultados obtenidos por su trabajo dependen de la calidad del proceso previo para la proposición de candidatos, ya que en ningún momento se persigue distinguir secuencias completamente carentes de exoplanetas de aquellas que sí lo hagan.

Según la metodología que presentan sus autores, es necesario realizar varios procesos de limpieza y preparación hasta adaptar las series temporales y grafos de flujo iniciales a un formato adecuado sobre el que aplicar modelos neuronales. Este preproceso de los datos consiste en concatenar las series temporales disponibles de una misma estrella, y buscar en la secuencia resultante fenómenos con las características de un tránsito planetario. En esta búsqueda aparecerán tanto tránsitos reales como aquellos producidos por otras fuentes, el objetivo final del modelo será hallar una forma de diferenciar estos dos casos y catalogar la estrella en consecuencia. Una vez localizados, se aíslan los posibles periodos de tránsito y son combinados en una única señal, "plegando"la señal original hasta que los distintos tránsitos repartidos de forma periódica coinciden en el mismo punto. Esta señal representa en un único segmento todos los posibles periodos de tránsito planetario de la estrella, representando lo que podría considerarse el aspecto medio del conjunto de todos los tránsitos. De la señal combinada de tránsitos se extraen entonces dos ventanas, una global y una local, diferenciadas simplemente por la cantidad de puntos de la secuencia que rodean al centro del evento de tránsito y son incluidos en la ventana. Estos puntos de vista globales y locales de cada estrella son los datos de entrada de los modelos neuronales con los que experimentaron Shallue y Vanderburg.

Los modelos desarrollados para este método requieren por lo tanto aceptar dos entradas simultáneamente, una para la vista global y otra para la local. La solución que se propone es una red basada en convoluciones unidimensionales con dos ramas, cuyos resultados se concatenan antes de pasar al segmento *Fully Connected* final del modelo. La salida del modelo viene dada por una simple capa densa con una neurona de salida y activación sigmoide, representando las dos probabilidades para la señal de entrada: la señal muestra la presencia de un exoplaneta o se trata simplemente de una señal negativa.

Este trabajo, con fecha de publicación en 2018, es considerado el mejor ejemplo del estado del arte en cuanto a la aplicación de modelos neuronales y ML a la búsqueda de exoplanetas, pero sus resultados están limitados por ciertas condiciones. En primer lugar, la búsqueda de periodos de tránsito para la extracción de las vistas global y local requiere conocer de antemano la frecuencia, profundidad y amplitud de los periodos que se espera encontrar en la señal de una estrella. Estos datos adicionales se obtienen de fuentes ajenas

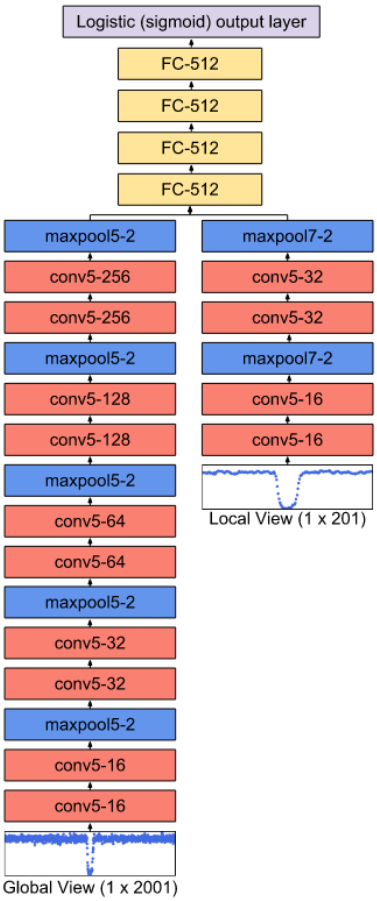

Figura 2.8: Arquitectura de la red Neuronal empleada por Shallue y Vanderburg. Los valores que acompañan a las capas convolucionales indican el tamaño de kernel y el número de filtros, los de las capas *Maxpool* indican el tamaño de *kernel* y el *stride*; y por último los valores de las capas *Fully Connected* indican el número de neuronas.

a los archivos donde podemos encontrar almacenadas las secuencias de flujos de luz registrados por los telescopios espaciales, y son en realidad obtenidas por la exploración inicial de la NASA junto al tratamiento de las secuencias en sus *pipelines*. Esto supone, no solo limitar el alcance de las muestras a las que se va a exponer al modelo, escogiendo solo aquellas que hayan sido propuestas como candidatas, si no que adicionalmente requiere información extraída de los procesos que se están intentando validar.

Aun considerando estas limitaciones, los resultados y la utilidad del modelo propuesto son innegables, alcanzando una precisión del 94.9 % en la catalogación de muestras recogidas por el telescopio *Kepler*. El código del modelo (bautizado como ASTRONET) y el *data set* empleado en este trabajo se encuentra accesible de forma pública en el repositorio GitHub de los autores[3].

Otra aplicación de los modelos neuronales a la clasificación de muestras de estrellas reales la encontramos en el trabajo de Hinners et al. [10], quienes aplican modelos neuronales recurrentes para resolver tanto problemas de clasificación como de regresión a partir de series temporales de flujos de luz, a la vez que comparan sus resultados empleando técnicas de *Machine Learning* clásicas. Al contrario que en el trabajo de Shallue y Vanderburg, los autores no realizan un procesamiento tan exhaustivo de las señales de flujo antes de emplearlas como entradas de sus modelos. Mientras que la metodología de Shallue et al. requería conocer de antemano varios aspectos de los datos contenidos en

[3]https://github.com/google-research/exoplanet-ml/tree/master/exoplanet-ml/astronet

las secuencias con las que trabajaban, como la duración o la intensidad de los periodos de tránsito, para poder aislar estos eventos y construir un nuevo *data set*, la metodología de Hinners et al. aplica simplemente estos cambios sobre las secuencias temporales:

- Reducir la cantidad de puntos de la secuencia a emplear. Durante la exploración inicial de los datos, los autores observaron que, de los aproximadamente 70.000 puntos por secuencia disponibles de cada estrella, era posible prescindir iterativamente de 9 de cada 10 puntos de la secuencia y seguir obteniendo curvas de flujo lo bastante representativas visualmente.
- Normalizar las secuencias de flujo. Los valores en formato absoluto de las secuencias contienen poca información por si mismos y son inconsistentes entre distintos periodos de observación de una misma estrella. Los autores emplean una normalización por la mediana de la secuencia y normalizan cada punto para centrar la secuencia en torno a cero.
- Eliminar valores extremos de los datos con una desviación superior a 2.5 veces la desviación estándar de la secuencia.
- Los autores emplean las secuencias obtenidas mediante el telescopio *Kepler*, las cuales ocasionalmente contienen valores perdidos. El último paso del preproceso será interpolar estos valores para rellenar cualquier hueco en la secuencia.

En cuanto a los modelos empleados, la primera aproximación de Hinners et al. consiste en el uso de una red neuronal recurrente basada en el uso de capas LSTM. La redes neuronales convolucionales carecen de la capacidad para tener en cuenta la secuencialidad de los datos que reciben como entrada; observan los datos como un todo y a partir de ellos extraen características que les ayudan a llevar a cabo tareas de clasificación o regresión. Por otro lado, las redes recurrentes basan su funcionamiento en la capacidad de 'recordar' el resultado de datos previamente observados, y aplicar ese conocimiento a los datos actuales. Para lograrlo las redes recurrentes reciben como entrada una secuencia de elementos; tras obtener el output de un elemento, la red almacena su estado y lo emplea para calcular el siguiente output del próximo elemento de la secuencia, repitiendo el proceso hasta completar la secuencia de entrada.

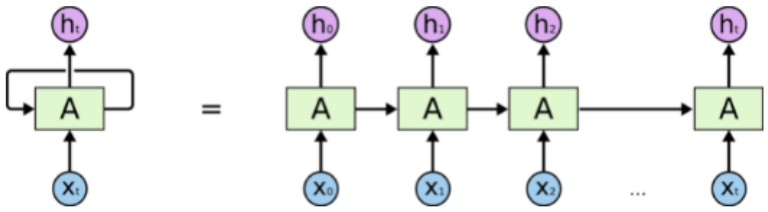

Figura 2.9: Esquema del funcionamiento de una red neuronal recurrente desplegada. La secuencia $(x_0, x_1, ..., x_t)$ seria la entrada de la red y $(h_0, h_1, ..., h_t)$ la salida

Las módulos LSTM son una ampliación de las redes recurrentes básicas orientada a solucionar un problema presente en este tipo de redes: las dependencias a largo plazo. Las redes recurrentes clásicas pierden la capacidad de relacionar dependencias entre distintos elementos de la secuencia de entrada a medida que estos elementos se encuentran más y más separados entre ellos. Las redes LSTM evitan este problema haciendo uso de varias redes neuronales en cada uno de sus módulos para decidir que información pasa o no al siguiente estado de la red a medida que se analiza la secuencia de entrada. Una explicación más detallada de estas redes y su funcionamiento puede encontrarse en el trabajo de Staudemeyer et al.[11]

De esta manera, las redes LSTM parecen más que adecuadas para resolver el problema de clasificación de variaciones de flujo, ya que se van a emplear secuencias temporales de datos en los que se pretende detectar patrones periódicos. Sin embargo en los resultados del trabajo de Hinners et al., los autores exponen no haber sido capaces de lograr más del 52 % de acierto al clasificar la secuencias de flujo en función de la presencia o no de un exoplaneta.

Para cerrar esta sección de nuestro trabajo hacemos mención al trabajo *A survey on machine learning based light curve analysis for variable astronomical sources* [12], en el que se realiza un análisis en profundidad del estado del arte en el ámbito del ML aplicado al estudio de secuencias de flujo.

CAPÍTULO 3
Fuentes de datos empleadas

En esta sección se detallan los datos que se han empleado, sus fuentes, su formato y el tratamiento que han recibido antes de ser adecuados para la experimentación con ellos.

3.1 *Mikulski Archive for Space Telescopes (MAST)*

Telescopios espaciales como *Hubble, Kepler* o *TESS* observan y toman mediciones de miles de cuerpos celestes durante años, generando una cantidad de datos considerable que necesita ser tratada y almacenada de forma ordenada y accesible. El *Space Telescope Science Institute (STSI)* es una institución que en 1991, con el lanzamiento del telescopio Hubble, comenzó a ofrecer apoyo a la misión realizando distintas tareas de soporte y que desde entonces ha continuado esta labor con las posteriores misiones de observación espacial de la NASA. Fue durante los primeros años del trabajo conjunto entre STSI y NASA que esta última decidió delegar la tarea de almacenar y conservar los resultados de las misiones de observación espacial en el instituto[8]. De esta manera, STSI creó en 1997 el *Multimission Archive for Space Telescopes (MAST)*, renombrado en 2015 a *Barbara Mikulski Archive for Space Telescopes*, un portal desde el que acceder a toda la información relativa a misiones espaciales dedicadas a la observación de radiación visible, infrarroja y ultravioleta[9].

De entre las misiones *Kepler*, *K2* y *TESS*, las tres misiones más representativas en la búsqueda de exoplanetas por parte de la NASA y accesibles desde el *Mikulski Archive*, se ha decidido trabajar en este proyecto únicamente con los datos disponibles de la misión *Kepler*, siguiendo los pasos de multitud de otras investigaciones similares. La razón detrás de esta elección es la siguiente: el entrenamiento de los modelos que vamos a desarrollar requiere muestras supervisadas, y dado que la disponibilidad de muestras negativas (sin presencia de exoplanetas) no supone un problema, debemos centrarnos en aquella fuente con mayor cantidad de exoplanetas confirmados. De los 4461 exoplanetas detectados y confirmados por la NASA, el 63 % lo fueron mediante las mediciones recogidas duarnte la misión *Kepler*.

3.1.1. Formato *Flexible Image Transport System*

Los datos de Kepler se componen por mediciones de flujo de 200.000 estrellas tomadas cada 30 minutos a lo largo de 4 años. Estas mediciones se encuentran segmentadas en trimestres de aproximadamente 90 días de duración, sumando un total de 18 trimestres y más de 70.000 observaciones totales por cada estrella. El número de observaciones en cada trimestre varía entre uno y otro debido, por ejemplo, a fallos técnicos del satélite *Kepler* que impidiesen tomar mediciones durante periodos de tiempo concretos o debido

a la corrupción de datos ya recogidos. Los datos se encuentran almacenados en el MAST en formato FITS y es posible acceder a su descarga a través de una API web, con la que podemos construir peticiones URLs para cada uno de los datos que requiramos; de esta API hablaremos más adelante. FITS [1] son las siglas para *Flexible Image Transport System*, un estándar abierto para definir el formato de archivos digitales dirigido a facilitar el almacenamiento, la transmisión y el procesamiento de datos. FITS es el formato mayormente empleado entre la comunidad astronómica, estando reconocido por la NASA y por la *International Astronomical Union (IAU)*. Concretamente para los archivos correspondientes a la misión Kepler, los archivos en formato FITS contienen multitud de diferentes datos relativos a la misión, no solamente las mediciones de flujo. Tal y como se describe en el *MAST Kepler Archive Manual*, adicionalmente a las observaciones de flujo, los distintos campos de este tipo de archivos incluyen información sobre el tiempo en el que se tomaron las mediciones, la cadencia, distintos *flags* indicando si se produjo algún error durante la medición, y las imágenes a partir de las cuales se extrajeron las secuencias, entre otros.

Los datos en crudo recibidos desde el telescopio *Kepler* incluyen una cantidad significativa de errores sistemáticos y estocásticos [15]. Estos errores pueden estar provocados por un amplio rango de motivos, como por ejemplo:

- Temblores. Periódicamente el satélite debe re-orientarse a medida que orbita alrededor de la Tierra y el Sol. Durante estos periodos, el movimiento del satélite puede afectar a los instrumentos, alterando las señales que registran.
- Cambios de temperatura. Las variaciones de temperatura provocan que los metales del satélite se expandan y contraigan, generando interferencia con los instrumentos a bordo.
- Variaciones de velocidad. Provocan que se pierda enfoque sistemáticamente sobre las estrellas observadas hasta que se aplique una re-orientación del satélite.

Estos errores son plenamente conocidos por el equipo de la misión *Kepler* y durante el post-proceso incluido en el *pipeline* de la misión se realiza un esfuerzo por corregirlos, de manera que las secuencias de datos resultantes solamente incluyan las variaciones propias de las estrellas monitorizadas. Los resultados de la corrección de estos errores los encontramos en el campo 'PDCSAP_FLUX' en los archivos FITS almacenados en el *Mikulski Archive*, y su uso para este tipo de investigación es consistente con la metodología observada en trabajos similares.

3.1.2. Descarga de los datos

El *Mikulski Archive* pone a nuestra disposición distintos métodos mediantes los que descargar los datos de las múltiples misiones a nuestro entorno local de trabajo, siendo uno de ellos una API web que nos permite construir peticiones URL para descargar únicamente aquellos elementos que deseemos. Uno de los trabajos que se han comentado en el Capítulo 2 es la investigación de Shallue Y Vanderburg sobre la aplicación de redes neuronales convolucionales a la detección de exoplanetas [4]. Aunque si bien nosotros hemos trabajado con un tratamiento completamente diferente de los datos, ambos trabajos parten de los mismos archivos base. Es por ello que hemos adaptado parte del código que podemos encontrar en el repositorio GitHub de los autores [2] para generar un fichero de descarga con las URLs de aquellos elementos del MAST que sean de interés

[1] https://fits.gsfc.nasa.gov/
[2] https://github.com/google-research/exoplanet-ml/tree/master/exoplanet-ml/astronet

para nuestro trabajo. Este código adaptado requiere como entrada un archivo CSV con la información de las diferentes estrellas de las que requerimos obtener sus secuencias de flujo, identificadas por el *Kepler Identification Number* (KepID) que se les otorgó durante el desarrollo de la misión *Kepler*.

3.2 *NASA Exoplanet Archive*

3.2.1. Selección de los datos

En el Capítulo 2 hemos hablado de como otro equipos y autores se han visto obligados a emplear datos simulados en sus trabajos para disponer de un *data set* de tamaño aceptable y, al mismo tiempo, compensar el número de muestras positivas y negativas. Estos trabajos suelen concatenar los datos de cada trimestre correspondiente a una estrella y emplear esa secuencia completa para el entrenamiento de distintos modelos. Sin embargo, dado que no disponemos de suficientes muestras positivas y una secuencia de ese tamaño posee una longitud de más de 70.000 puntos individuales, la cual requeriría más capacidad de computo para poder aplicar una red neuronal de la que tenemos disponible, hemos decidido interpretar las secuencias de cada trimestre por separado como secuencias completas. Al mismo tiempo que obtenemos ventajas como aumentar el número de muestras a las que tenemos acceso empleando el mismo *data set* inicial; esta metodología también ha generado ciertos inconvenientes que hemos afrontado con un preproceso de las señales más extenso y nuevas fuentes de datos.

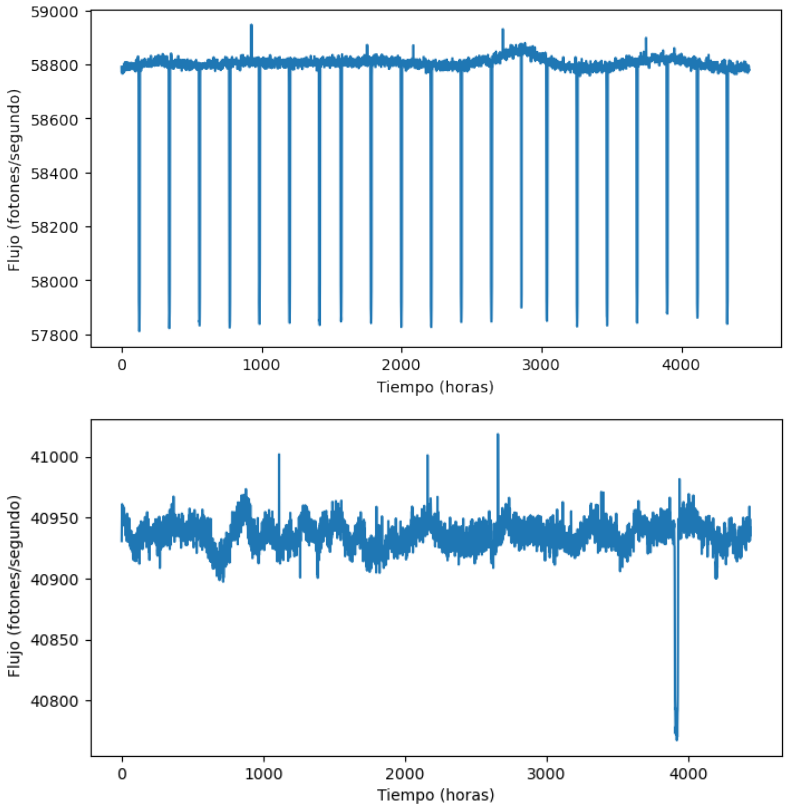

Figura 3.1: Comparativa gráfica de secuencias de flujo para exoplanetas con distinto periodo orbital. En la gráfica superior podemos ver como con un exoplaneta con periodo orbital de 4 días se producen múltiples bajadas en los valores de flujo en un trimestre completo. Por el contrario, en la gráfica inferior se representa la curva de flujo de una estrella con un exoplaneta de periodo orbital igual a 331 días.

La principal desventaja de trabajar con secuencias por trimestre radica en que ya no es posible emplear la totalidad de las muestras positivas confirmadas. El objetivo que nuestros modelos deben alcanzar pasa por aprender a reconocer las bajadas periódicas en la intensidad del flujo que aparecen en las secuencias, pero si la periodicidad de dichas bajadas es mayor a la longitud de la secuencia que los modelos reciban como entrada esta tarea pasa a ser imposible. La duración aproximada de un trimestre es de 91 días, de manera que inicialmente será necesario descartar todos aquellos exoplanetas confirmados hasta la fecha con un periodo orbital superior a esta duración. Aun así, no es suficiente con que se produzca una única bajada del flujo por trimestre, nuestros modelos necesitan encontrar la relación de periodicidad entre estas bajadas causadas por exoplanetas, para distinguirlas de aquellas bajadas esporádicas motivadas por otras fuentes. Hemos decidido que al menos deben estar presentes 2 o 3 de estas bajadas por cada periodo de 90 días, por lo tanto hemos decidido emplear tan solo aquellos exoplanetas confirmados con un periodo orbital inferior a 30 días. En la figura 3.1 queda ilustrado de forma visual este problema: mientras que un exoplaneta con periodo orbital de unos pocos días genera múltiples descensos del flujo, uno con un periodo de varios cientos de días lo hará como máximo una vez por trimestre, haciendo imposible su detección si disponemos exclusivamente de este periodo de 90 días.

3.2.2. *NASA Exoplanet Archive*

Para seleccionar los datos de aquellos exoplanetas que cumplan los requisitos que hemos establecido para nuestra experimentación, es necesario explorar y filtrar los datos de exoplanetas confirmados. Esta funcionalidad la hemos encontrado en el *NASA Exoplanet Archive*[16], un catalogo y base de datos en linea donde se enlazan los datos de diferentes exoplanetas y sus estrellas anfitrionas. Esta herramienta fundada por la NASA incluye tablas interactivas con propiedades para todos los exoplanetas confirmados, así como listas de candidatos a exoplaneta o información individual de periodos de tránsito detectados, entre otras funcionalidades. Los responsables de este archivo actualizan la información contenida en este semanalmente, monitorizando los artículos publicados en diferentes medios de divulgación aceptados e incluyendo los parámetros de nuevos planetas descubiertos que en ellos se presenten.

Como ya hemos explicado, para la descarga de los datos se ha adaptado parte del código del modelo *ASTRONET*, el cual requiere que generemos de antemano un CSV con la información de las estrellas en las que deseemos centrarnos. Para acceder a los KepID de estas estrellas se ha hecho uso de las tablas interactivas disponibles en el *NASA Exoplanet Archive*; en concreto la tabla correspondiente a los periodos de tránsito detectados en las observaciones de la misión *Kepler* entre los trimestres 1 y 17[3]. En esta tabla encontramos parámetros tanto de los exoplanetas ya descubiertos, como de aquellos aun considerados como candidatos a la espera de ser confirmados, o de aquellos ya etiquetados como falsos positivos. La tabla nos permite filtrar los datos según distintos campos y parámetros, incluyendo una columna que ha resultado de considerable utilidad en este trabajo. La columna *'Autovetter Training Set Label'* de la tabla etiqueta cada una de filas en una de 4 posibles categorías:

- PC (*Planet Confirmed*). Exoplanetas confirmados.
- AFP (*Astrophysical False Positive*). Falsos positivos, periodos de tránsito generados por otros fenómenos.
- NTP (*Non-Transiting Phenomenon*). No existe periodo de tránsito.

[3] https://exoplanetarchive.ipac.caltech.edu/cgi-bin/TblView/nph-tblView?app=ExoTbls&config=q1_q17_dr24_tce

- UNK (*Unknown*). Se desconoce la causa de las bajadas en el flujo detectado.

Haciendo uso de estas etiquetas y de la columna '*Orbital Period*' hemos generado todos los CSV necesarios para la descarga de los datos desde el MAST. Hemos dividido los datos en 3 lotes:

- **Datos Positivos**. Se incluye en este CSV los KepID de todas aquellas estrellas anfitrionas de al menos un exoplaneta confirmado con un periodo orbital inferior o igual a 30 días.
- **Datos Negativos: Falsos Periodos de Tránsito**. Se incluye en este CSV los KepID de todas aquellas estrellas que en algún punto pasado se consideró anfitrionas de un exoplaneta con un periodo orbital inferior o igual a 30 días, pero que con el tiempo se ha demostrado que no existe tal cuerpo.
- **Datos Negativos: Sin Periodos de Tránsito**. Para este caso hemos seguido otro planteamiento. Dado que la mayoría de datos disponibles en el MAST pertenecen a estrellas sin presencia de exoplanetas, se ha generado una lista de KepIDs de estrella aleatorios, los cuales se han filtrado en función de los CSV generados para las dos clases anteriores. Si alguno de los ID pertenece a los Positivos o a los Falsos Periodos de Tránsito se elimina de la lista de Negativos.

Una vez generados los CSV podemos proceder a la descarga de los archivos FITS desde MAST, los cuales quedarán ordenados en carpetas según el ID de la estrella de origen de cada medición.

3.3 Sumario: Fuentes de datos

En resumen, el proceso seguido para obtener los datos necesarios empieza obteniendo las listas de KepIDs de las estrellas con cuyas secuencias de flujo vamos a trabajar (descartando aquellas con un periodo orbital demasiado grande), usando para conseguirlo las tablas interactivas disponibles en el *NASA Exoplanet Archive*. A partir de estas IDs y haciendo uso de fragmentos de código adaptados desde el proyecto *ASTRONET*, descargamos los ficheros FITS que contienen las secuencias desde el *Mikulski Archive for Space Telescopes*. Como resultado, disponemos de un total de 76163 secuencias, a las cuales debemos aplicar un preproceso antes de ser adecuadas para el entrenamiento de los modelos neuronales.

CAPÍTULO 4
Preproceso de los datos

En este apartado se encuentran redactadas las modificaciones o ampliaciones que han sido necesarias aplicar a las secuencias de valores de flujo bases antes de pasar al entrenamiento de los modelos.

4.1 Valores perdidos e interpolación

Reiteramos en esta sección que las mediciones obtenidas por el satélite/telescopio *Kepler* se hallan condicionadas en gran medida por las limitaciones tecnológicas del mismo. La mayoría de las secuencias cuatrimestrales registradas por *Kepler* contienen algún tipo de interferencia como causa de ello, aunque como ya hemos comentado, al estar documentadas estas anomalías pueden ser corregidas al recibir las secuencias aquí en la Tierra aplicando un *pipeline* adecuado. Por desgracia, a lo largo de los 4 años en los que el satélite estuvo en funcionamiento se produjeron ciertos eventos y fallos técnicos que impidieron registrar ningún dato durante considerables periodos de tiempo, culminando con el fallo en 2013 de uno de los componentes clave del satélite, que puso fin a la misión.

Antes de realizar cualquier tipo de preproceso, deberemos filtrar pues aquellas secuencias cuatrimestrales que no contengan una cantidad suficiente de valores observados. Inicialmente, la programación de la misión estipula que el satélite debe tomar una medición de cada una de las estrellas observadas cada 30 minutos, por lo que en un trimestre de 91 días deberían aparecer en el mejor de las casos secuencias con un total aproximado de 4300 mediciones. Como condición adicional, los modelos basados en redes neuronales que vamos a emplear requieren que todas las secuencias posean la misma longitud, por el contrario las secuencias de cada estrella y trimestre poseen longitudes variables a causa de fallos y datos perdidos. En base a estos hechos se ha tomado la decisión de filtrar aquellas series que no superen un mínimo de observaciones y, para aquellas que si lo superen, recortar las secuencias a una longitud fija.

Por otra parte, también es posible que los datos perdidos como consecuencia de diferentes causas no ocupen largos periodos de tiempo y se limiten simplemente a observaciones puntuales. Para este caso, en el código de lectura y extracción de las secuencias desde los ficheros FITS se ha incluido una función de interpolación para recuperar esos valores perdidos.

Originalmente, tras la descarga de los ficheros disponíamos de 76163 secuencias, si embargo aproximadamente un 60 % de ellas no superaban los filtros que hemos establecido, por lo que finalmente nuestro *data set* se compone de 45511 secuencias. El reparto de estas secuencias según su etiqueta en el *NASA Exoplanet Archive* puede observarse en Figura 4.1 y Figura 4.2.

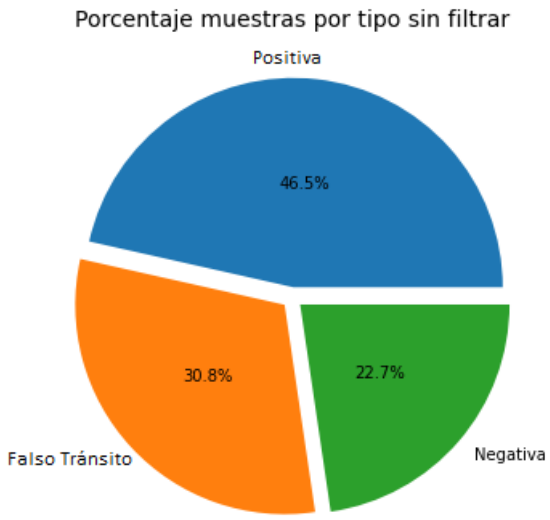

Figura 4.1: De un total de 76163 muestras descargadas, en el gráfico se muestra el porcentaje de cada tipo de estas.

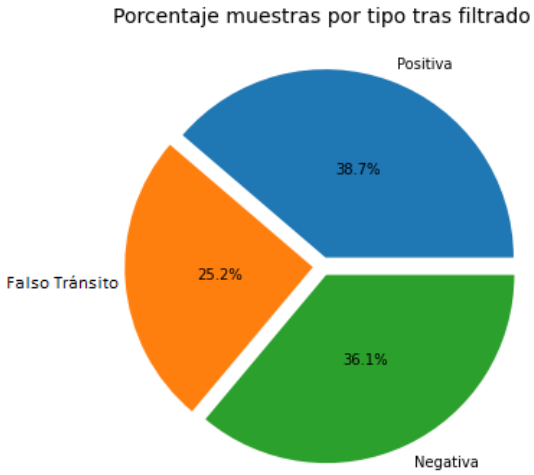

Figura 4.2: Tras filtrar las muestras y eliminar aquellas no validas el total de muestras se reduce a 45511, repartidas tal y como se muestra en el gráfico.

4.2 Etiquetado y división de las muestras

Una vez filtradas y extraídas las secuencias de los ficheros FITS, estas se han organizado en matrices donde cada fila corresponde a una secuencia de 4200 puntos. A cada matriz se le ha incorporado una nueva columna según el tipo de secuencia indicando su etiqueta: las muestras positivas se han etiquetado con el valor '1', mientras que las muestras negativas y falsas positivas se han etiquetado con el valor '0'. Dichas etiquetas serán las que proporcionaremos a nuestros modelos neuronales junto con las secuencias para resolver el problema de clasificación binaria.

Es necesario distribuir las muestras entre grupos de entrenamiento, validación y test antes de proceder a la experimentación. Dado que estamos tratando cada trimestre como una secuencia completa, tenemos potencialmente hasta 17 posibles secuencias pertenecientes a una misma estrella. Si dividiésemos las muestras de forma aleatoria, muestras pertenecientes a una misma estrella aparecerían tanto en los conjuntos de test, como en

los de entrenamiento y validación; dada la similitud entre estas secuencias (por proceder de la misma fuente a pesar de ser distintas) ocasionarían un aumento artificial de la precisión de nuestros modelos. Para evitar esta situación, mantenemos siempre las secuencias originadas por cada estrella en la misma segmentación que el resto de sus secuencias. Teniendo este requisito en cuenta, el resto del reparto se produce aleatoriamente, destinan un 20 % de las muestras a test y el resto a entrenamiento; de entre las muestras de entrenamiento otro 20 % de estas se destina a validación.

4.3 Normalización y *Data Augmentation*

Hasta ahora hemos tratado las secuencias como un todo, sin examinar los valores que contienen ni aplicar ningún tratamiento sobre estos. Las mediciones tomadas por el satélite *Kepler*, incluso para una misma estrella, varían sus valores medios entre una secuencia y otra en un orden de magnitud de hasta 10^2 unidades. Esto ocurre debido a que, tras completar un trimestre, el satélite debe re-orientarse para seguir observando la misma porción del cielo nocturno, lo que provoca un cambio en las lecturas recibidas a causa de su alta sensibilidad. Por otra parte, los valores absolutos de flujo (medidos en fotones por segundo) son demasiado elevados para poder trabajar con ellos de manera adecuada en una red neuronal. De emplear las secuencias manteniendo los valores de flujo absolutos, nos arriesgamos a que la red neuronal requiera excesivo refinamiento por nuestra parte o a necesitar emplear un proceso de entrenamiento más lento hasta converger a una solución. Resolver este problema es simple, es necesario normalizar cada una de las secuencias hasta que todos los datos se encuentren en el mismo rango de valores. En la literatura que hemos explorado se propone normalizar los datos aplicando una normalización en torno a la mediana de los datos, sin embargo, durante nuestra experimentación se ha llegado a la conclusión de que normalizar los datos desplazando los datos de acuerdo a su media y empleando un*Min-Max Scaler*. De esta manera, cada una de las secuencias de flujo se ha normalizado hasta que sus valores se encuentren centradas en su media, entre el rango 1 y -1.

Otro de los problemas más comunes, sobre todo en situaciones como las que nos plantea este trabajo, donde los datos son simplemente secuencias temporales de una dimensión o cuando el tamaño del *data set* con el que se trabaja es reducido, es el sobreentrenamiento de los modelos neuronales. En ciertas ocasiones, las redes neuronales pueden aprender a distinguir las muestras de entrenamiento y validación a nivel individual, en lugar de reconocer aquellas características propias de las clases con las que dichas muestras están etiquetadas. Esta situación provoca que durante el entrenamiento se alcancen valores de precisión en la clasificación de las muestras demasiado optimistas, mientras que al aplicar la misma red a los datos de test, a los cuales el modelo no ha tenido acceso antes, normalmente la precisión desciende notablemente. Para evitar esta situación, una posible solución se encuentra en aplicar un proceso de *data augmentation*.

Data augmentation hace referencia al conjunto de operaciones y técnicas que podemos aplicar sobre nuestro *data set* para 'aumentar' artificialmente el número de muestras a las que tenemos acceso, obteniendo versiones ligeramente modificadas de las originales. Las técnicas que podemos aplicar dependen del formato de los datos que estemos empleando, por ejemplo: si nos encontramos con una tarea de clasificación de imágenes podemos invertir, rotar o recortar las imágenes originales del *data set* para obtener nuevas imágenes. En nuestro caso, dado que vamos a trabajar con secuencias temporales de una dimensión, hemos optado por implementar los siguientes métodos de *data augmentation*:

- Invertir. La forma más simple de aumentar este tipo de datos es simplemente emplear la misma secuencia pero a la inversa. De esta manera las relaciones de distancia entre los descensos periódicos de flujo se mantienen a la vez que obtenemos nuevos valores para la secuencia. El problema es que esta técnica es muy limitada, pues solo nos permite obtener una nueva secuencia por cada una de las que ya disponemos.
- Desplazamiento lateral. Podemos interpretar las secuencias como señales cerradas cíclicas, de forma que tras el último valor de la secuencia volveríamos a empezar desde el primero. Con esta interpretación podemos aplicar una técnica de *data augmentation* en la que seleccionemos segmentos de longitud aleatoria al final o comienzo de la secuencia, cortemos dichos segmentos de su posición original y los insertemos en el extremo opuesto de la secuencia.
- Selección de ventanas. Alternativamente a trabajar con las secuencias cuatrimestrales completas podemos extraer ventanas de menor tamaño de estas. De esta forma, seleccionando aleatoriamente una de estas ventanas en cada paso del entrenamiento seriamos capaces de extraer múltiples muestras de una única secuencia.
- Escalado. Tras normalizar los datos, podemos re-escalar de forma aleatoria los valores de una secuencia en un rango controlado.

Estas distintas técnicas, que pueden aplicarse en conjunto o por separado, las aplicaremos en cada *epoch* del entrenamiento de los modelos. Es decir, no calcularemos las nuevas muestras previamente a iniciar el entrenamiento, aumentando realmente el número de muestras a emplear; si no que partiendo de las muestras reales, en cada *epoch* del entrenamiento emplearemos una versión modificada aleatoriamente mediante las técnicas de *data augmentation* que hemos presentado. De esta forma mantendremos el mismo tamaño de *data set* a la vez que disponemos de acceso a nuevas muestras. Para lograrlo será necesario implementar un *data generator*.

Otros métodos para solucionar el sobre-entrenamiento son el uso de las capas *dropout* en la arquitectura de las redes neuronales. Tanto de los detalles del *data generator*, como de la inclusión de capas *dropout* hablaremos en detalle en el siguiente capítulo.

CAPÍTULO 5
Redes Neuronales y experimentación

En este capítulo vamos a describir la arquitectura de los modelos neuronales que hemos desarrollado, su origen y como han evolucionado durante la experimentación hasta alcanzar su estructura definitiva.

Para el diseño de nuestras redes neuronales y procesos de *data augmentation* hemos hecho uso de Keras [17], una API para lenguaje Python construida sobre Tensorflow2 [18], cuyo objetivo radica en simplificar las acciones necesarias por el usuario y reducir la carga de trabajo necesaria para construir y entrenar diferentes modelos neuronales. Por otro lado, para la ejecución de nuestro código hemos hecho uso del entorno virtual *Google Colaboratory*, el cual nos permite instalar librerías de forma autónoma y ejecutar *scripts* Python, al mismo tiempo que nos permite emplear aceleración del código mediante el uso de GPU (*Graphic Processing Unit*), un valor añadido realmente importante a la hora de entrenar modelos neuronales pues reduce de forma considerable el tiempo de ejecución necesario.

5.1 *Data Generator*

Este elemento de nuestro código será el responsable de aplicar las distintas técnicas de *data augmentation* que se han descrito en el capítulo anterior. Para entender su funcionamiento debemos comprender antes como Keras entrena los modelos neuronales. Al lanzar el entrenamiento de un modelo, Keras recibe como parámetros, entre otros datos, el conjunto de datos de entrenamiento y sus etiquetas. Tras recibir los datos, estos se agrupan en *batches*, subconjuntos del total de muestras de entrenamiento, que se emplean para definir cada cuantas muestras analizadas deben actualizarse los pesos del modelo. Una vez el modelo ha empleado todos los *batches*, es decir, ha realizado una pasada completa sobre cada una de las muestras de entrenamiento, el modelo finaliza un *epoch*.

Al aplicar *data augmentation*, nuestro *data generator* será el que reciba el conjunto de entrenamiento y sus etiquetas como argumentos y, pasaremos una instancia de este como argumento al modelo. De esta manera, cuando el modelo solicite un *batch* del conjunto de entrenamiento, nuestro *data generator* deberá seleccionar aquellas muestras que correspondan al *batch* solicitado y construir un nuevo *batch* modificando aleatoriamente dichas muestras mediante diferentes técnicas de *data augmentation*. Cada vez que el modelo solicite un mismo *batch* recibirá pues un conjunto de muestras diferentes a causa de las modificaciones aleatorias, pero el conjunto original de entrenamiento nunca llegará a modificarse y se mantendrá intacto en el *data generator*. Una vez finalizado un *epoch*,

nuestro *data generator* barajará las muestras de entrenamiento de forma que estas sean proporcionadas al modelo en un orden diferente en cada *epoch*, ayudando así a reducir el sobre-entrenamiento.

En resumen, el modulo *data generator* que hemos implementado recibirá un conjunto de datos y sus etiquetas, y será capaz de aplicar una, ninguna o varias técnicas de *data augmentation*, según los parámetros empleados, siempre reordenando las muestras al finalizar un *epoch*. Nuestro *data generator* ha sido empleado para generar tanto los conjuntos de entrenamiento como los de validación, pero no se ha aplicado al conjunto de test.

5.2 Red recurrente LSTM

La primera de las arquitecturas neuronales con las que se ha experimentado han sido las redes recurrentes. Como ya se ha expuesto en el Capítulo 2, este tipo de redes reutilizan los estados pasados de la red y aplican esa información a posteriores decisiones, lo que resulta de utilidad en la clasificación de series temporales. Para nuestro diseño y experimentación con redes LSTM hemos partido de la base propuesta por el trabajo de Hinners et al. [10]; a pesar de que en el artículo publicado los resultados empleando este tipo de redes a penas consiguen superar el 50 % de precisión al ser aplicadas a nuestro problema, en el mismo artículo se reflexiona como esto podría ser debido a un problema con los datos empleados y no a causa de la arquitectura neuronal empleada.

5.2.1. Diseño de la red

Nuestra red recibe como entrada las secuencias cuatrimestrales de valores de flujo con una longitud de 4200 valores, sin embargo, las capas LSTM requieren dividir la secuencia en segmentos que puedan ser analizados de forma secuencial, de manera que la información extraída de los primeros segmentos pueda ser empleada para seguir analizando los segmentos posteriores. Tras experimentar con varios valores, se ha encontrado que la división que ofrece mejores resultados se obtiene al segmentar la secuencia inicial en 42 segmentos con 100 valores cada uno.

Hemos empleado capas LSTM bidireccionales frente a las capas LSTM convencionales ya que, debido a la naturaleza de las señales con las que estamos trabajando, los patrones que deseamos que la red sea capaz de distinguir deberían de ser detectables al analizar las secuencias tanto en orden cronológico como en orden inverso. Si bien conocemos la estructura que debería de seguir una red de capas LSTM destinada a la clasificación de series temporales, no conocemos qué cantidad de capas necesitamos implementar, ni qué cantidad de unidades de salida debemos parametrizar para cada una ellas de manera que obtengamos resultados óptimos.

Como sección de salida de la red empleamos dos capas *fully connected*, una de ellas con la misma cantidad de neuronas que la salida de la última capa LSTM y una capa de salida con una única neurona y activación sigmoide, de forma que el *output* de la red siempre se corresponda con un valor entre 0 y 1, siendo cualquier muestra con un valor de salida superior a un limite considerada positiva y para cualquier valor opuesto considerada negativa. En la Figura 5.1 podemos observar la estructura de una de las posibles redes con está arquitectura, compuesta por 3 capas LSTM con 32 filtros cada una.

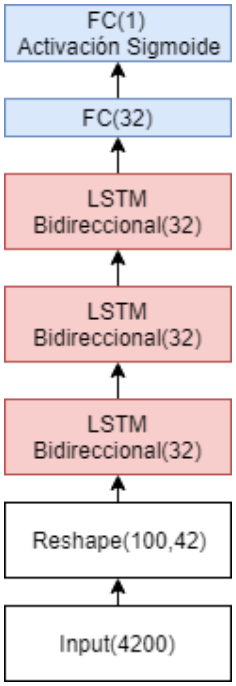

Figura 5.1: Diagrama mostrando la arquitectura LSTM empleada. Los valores que acompañan a las capas FC representan el número de neuronas que las componen; los que acompañan a las capas LSTM indican el número de unidades de salida; y los valores de las capas *Reshape* e *Input* representan las dimensiones de la muestra.

5.2.2. Experimentación

Durante la fase de experimentación ha sido necesario modificar la arquitectura de nuestra red hasta hallar la combinación de parámetros y capas capaz de alcanzar el valor de precisión óptimo durante la clasificación de las muestras de test. En esta memoria no se incluirán los resultados de todas las combinaciones de hiper-parámetros con las que se ha experimentado, solamente de aquellas relativas al número y tamaño de las capas empleadas en la red neuronal. Los resultados de cada combinación de capas se incluyen en el capítulo siguiente, sin embargo podemos detallar otros de los parámetros aplicados a nuestra red, cuyos valores hemos hallado mediante previa experimentación:

- *Learning Rate*. Este parámetro controla la magnitud con la que el modelo actualiza sus pesos al finalizar cada *batch* de muestras. Un *learning rate* demasiado elevado puede condicionar el modelo a converger a soluciones excesivamente generales y de poca precisión; por otro lado, un *learning rate* demasiado bajo puede ralentizar el aprendizaje del modelo e impedir que supere mínimos locales hasta alcanzar la solución óptima. Durante la experimentación se ha concluido que el valor adecuado para estos modelos debe fijarse en 0.001 durante todo el entrenamiento, puesto valores superiores no ofrecen resultados positivos y valores inferiores generan un problema de sobre-entrenamiento.

- Optimizador. La API Keras nos ofrece implementaciones de diferentes optimizadores con los que compilar nuestro modelo. Tras varias repeticiones del entrenamiento se ha alcanzado la conclusión de que un optimizador Adam resulta apto para esta tarea de clasificación frente al resto de optimizadores.

- *Batch Size*. El tamaño de los distintos *batches* en los que dividimos los conjuntos de entrenamiento y validación se ha fijado en 64.

- El resto de parámetros no especificados en esta memoria corresponden a los valores por defecto de las distintas implementaciones de capas y elementos que nos ofrece Keras.

5.3 Red convolucional 1D

5.3.1. Diseño de la red

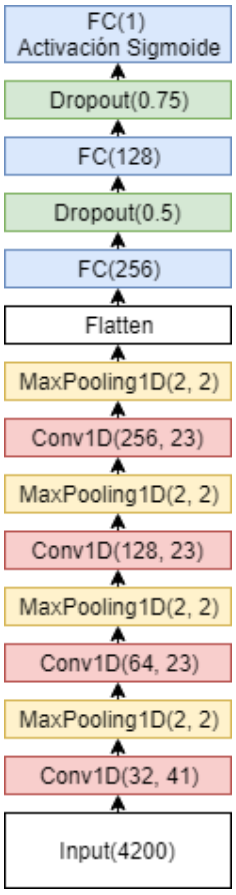

Figura 5.2: Diagrama mostrando la arquitectura Convolucional 1D empleada. Los valores que acompañan a las capas FC representan el número de neuronas que las componen; los valores de las capas *dropout* indican el porcentaje de conexiones aleatorias que son ignoradas en un paso concreto del entrenamiento; los valores de las capas *MaxPooling* indican respectivamente el tamaño de ventana y el *stride*; por último, los valores de las capas Conv1D representan respectivamente el número de filtros y el tamaño del kernel

Para las redes basadas en este tipo de capas neuronales hemos utilizado como base el trabajo de Jiang et al. quienes proponen una red compuesta por 4 capas convolucionales. Sin embargo, la arquitectura empleada en su trabajo no hace uso de ciertas técnicas propias de las redes convolucionales que nosotros hemos incorporado a su red, consiguiendo ciertas mejoras en los resultados frente a la red de partida. Estas mejoras incluyen:

- Capas *MaxPooling*. En la red original de Jiang et al. el tamaño de las secuencias se reduce en cada nivel de la red empleando un *stride* mayor de 1 en cada una de las capas convolucionales. Nuestro modelo sustituye este método mediante el uso de capas *MaxPooling*.

- Capas *Dropout*. Como ya hemos mencionado en varias ocasiones, uno de los principales inconvenientes que hemos encontrado durante el entrenamiento de los modelos es el sobre-entrenamiento de nuestras redes. Como soporte a las técnicas de *data augmentation* que ya se han descrito, hemos empleado capas *dropout* en el último segmento de nuestras redes convolucionales. Estas capas ignoran aleatoriamente un porcentaje de las conexiones en cada paso del entrenamiento, forzando al modelo a encontrar unos pesos que resuelvan el problema de forma general, sin depender de memorizar las muestras concretas que recibe durante el entrenamiento.
- Descenso exponencial del *learning rate*. A diferencia de nuestra red LSTM, donde esta técnica no presentaba ninguna ventaja, nuestros modelos convolucionales se han compilado con control del *learning rate*, de manera que este descienda de forma exponencial a medida que avanza el entrenamiento. Esto permite a la red afinar con precisión los valores de sus pesos.

5.3.2. Experimentación

Varias configuraciones y modificaciones de la red base se han empleado en la experimentación. Algunas de estas modificaciones incluían diferentes tamaños de kernel en las capas convolucionales, o el uso de arquitecturas de mayor complejidad, como la estructura *inception*, que nos permitía emplear varios tamaños de kernel simultáneamente en los distintos niveles de la red. Tras realizar distintas iteraciones de entrenamiento sobre estas modificaciones, la red con mejores resultados es la ilustrada en la Figura 5.2.

Al igual que en el caso de la red LSTM, se ha empleado un optimizador Adam y el resto de parámetros no nombrados se han mantenido en sus valores por defecto.

5.4 Red convolucional 2D

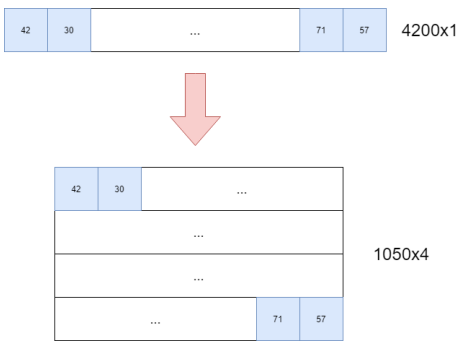

Figura 5.3: Visualización del efecto de la capa *reshape*, transformando una secuencia de dimensiones 4200x1 en una de 1050x4

5.4.1. Diseño de la red

También en el trabajo de Jiang et al. se propone transportar el problema de la clasificación de secuencias temporales al campo de la clasificación de imágenes, aplicando previamente al entrenamiento de las redes convolucionales una transformación a la secuencia. Esta es una aplicación poco ortodoxa las redes convolucionales en 2 dimensiones, pero de acuerdo a los resultados publicados por estos autores podría ser capaz de igualar los resultados de las convoluciones 1D.

Figura 5.4: Diagrama mostrando la arquitectura Convolucional 2D empleada. Los valores que acompañan a las capas FC representan el número de neuronas que las componen; los valores de las capas *dropout* indican el porcentaje de conexiones aleatorias que son ignoradas en un paso concreto del entrenamiento; los valores de las capas *MaxPooling* indican respectivamente el tamaño de ventana y el *stride*; por último, los valores de las capas Conv2D representan respectivamente el número de filtros y el tamaño del kernel

Para poder aplicar redes convolucionales en 2 dimensiones en necesario transformar las secuencias de flujo unidimensionales, empleando para ello una capa *reshape* al inicio de la red. Esta capa reorganiza la secuencia de entrada para modificar sus dimensiones a aquellas incluidas como parámetro. En nuestro caso, las series de flujo son unidimensionales, por lo que poseen una dimensión inicial de 4200x1; para tratar la secuencia como una imagen debemos redistribuir los puntos de la muestra de forma que se extiendan en ambas dimensiones, siempre manteniendo el mismo número de puntos en la muestra. Algunos ejemplos de dimensiones de salida serían 1050x4 (Figura 5.3) o 105x40.

Previamente a alcanzar la configuración final de la red se ha experimentado con distintos parámetros para la capa *reshape*. Sin embargo, no se ha encontrado una única combinación de dimensiones de salida definidas por la capa *reshape* que resulte en una clasificación óptima. En lugar de emplear una única transformación, la red se ha divido en tres ramas, cada una de ellas manejando una 'imagen' de diferente dimensión construida a partir de la misma secuencia inicial. Los resultados de las distintas ramas de la red tras el último bloque convolucional vuelven a transformarse en una secuencia unidimensional y son concatenados. El segmento *fully connected* de esta red es idéntico a la red basada en convoluciones 1D. En la Figura 5.4 es posible observar la estructura final del modelo.

5.4.2. Experimentación

En este caso y debido a su similitud, se ha procedido de forma idéntica al entrenamiento de la red convolucional 1D: se han realizado varias repeticiones con distintas modificaciones en el número de capas y ramas, así como variando el tamaño de los kernels en las capas convolucionales antes de seleccionar el modelo representado en la figura.

5.5 Métricas empleadas

En el Capítulo 3 se detallan las características de nuestro *data set*, el cual se divide en muestras positivas o negativas. El número de muestras en cada clase no se encuentra equilibrado, superando en gran número las muestras negativas a las positivas. A causa de esta diferencia, es necesario emplear métricas no afectadas por esta para comparar los resultados obtenidos por cada modelo.

5.5.1. *Accuracy*

Esta métrica representa el porcentaje de muestras correctamente clasificadas frente al total de muestras. De usar únicamente esta métrica, los resultados no serían completamente honestos, pues como ya hemos comentado existe una desigualdad en el tamaño de las clases. En el escenario en el que nuestro modelo clasificase cada muestra que recibe como elemento de la clase mayoritaria, el *accuracy* seguiría por encima del 50 %, pero no sería un resultado representativo de la capacidad del modelo para distinguir ambas clases. A pesar de ello, está métrica puede resultar de utilidad en la comparación de resultados, por lo que se ha mantenido en nuestros resultados.

$$accuracy = \frac{muestras\ correctamente\ clasificadas}{muestras\ totales} \tag{5.1}$$

5.5.2. *Precision*

En el contexto de las tareas de clasificación binaria, la *precision* es calculada de acuerdo a los siguientes resultados del modelo:

- *True positive* (TP). Muestras positivas correctamente clasificadas por el modelo
- *True Negative* (TP). Muestras negativas correctamente clasificadas por el modelo
- *False positive* (TP). Muestras erróneamente clasificadas como positivas.
- *False positive* (TP). Muestras erróneamente clasificadas como negativas.

Siguiendo esta terminología, la *precision* es calculada como la cantidad de muestras positivas correctamente calculadas entre el total de muestras clasificadas como positivas:

$$precision = \frac{TP}{TP + FP} \tag{5.2}$$

5.5.3. *Recall*

De forma similar a *precision*, la métrica *recall* se calcula en base a las muestras positivas correctamente clasificadas entre las muestras positivas totales:

$$recall = \frac{TP}{TP + FN} \tag{5.3}$$

5.5.4. Curva ROC y AUC

Habitualmente, en un problema de clasificación binaria, el umbral que decide si una muestra es aceptada como positiva o rechazada como negativa a partir del output de un modelo está fijo. Por ejemplo, nuestras redes neuronales poseen una capa salida con función de activación sigmoide, de manera que los resultados para las distintas muestras solo pueden encontrarse en el rango [0,1] y el umbral que separa ambas clases es el punto medio de este rango: si el resultado es superior a 0.5 la muestra es considerada positiva, si no lo es, se clasifica como negativa. Sin embargo, puede resultar de utilidad observar como el resultado de la clasificación va cambiando a medida que modificamos el umbral que separa ambas clases. La curva ROC toma la forma de una gráfica que representa la relación entre el ratio muestras positivas correctamente clasificadas frente al ratio de muestras negativas erróneamente clasificadas para distintos valores del umbral [19]. En primer lugar, será necesario calcular los ratios en función de los resultados parciales que se han descrito anteriormente.

$$True\ Positive\ Rate\ (TPR) = \frac{TP}{TP + FN} \tag{5.4}$$

$$False\ Positive\ Rate\ (FNR) = \frac{FP}{FP + TN} \tag{5.5}$$

Estos dos ratios deberán calcularse para todos los posibles umbrales en los que el número de muestras clasificadas en una clase u otra cambie, representando los resultados en una gráfica. Cada punto de la gráfica corresponderá a un punto formado con el valor de cada ratio, los cuales se unen para formar la curva. En la Figura 5.5 se ilustra un ejemplo de esta representación con 20 muestras y sus resultados de clasificación.

El uso de la curva resulta de ayuda para visualizar gráficamente los resultados pero no es adecuado para comparar distintos modelos, puesto que resultaría complicado equiparar dos curvas con una trayectoria similar. Para comparar curvas ROC de manera objetiva es posible hacer uso de una nueva métrica, el 'área bajo la curva' (Area Under Curve, AUC). Como su nombre indica, esta métrica simplemente calcula la superficie del grafo que queda bajo la curva ROC obtenida, siendo la superficie completa de todo el espacio de representación igual a 1. Este valor puede interpretarse como la probabilidad de que a una muestra positiva aleatoriamente escogida se le otorgue un mayor valor por el clasificador que a una muestra negativa aleatoriamente seleccionada.

Inst#	Class	Score	Inst#	Class	Score
1	p	.9	11	p	.4
2	p	.8	12	n	.39
3	n	.7	13	p	.38
4	p	.6	14	n	.37
5	p	.55	15	n	.36
6	p	.54	16	n	.35
7	n	.53	17	p	.34
8	n	.52	18	n	.33
9	p	.51	19	p	.30
10	n	.505	20	n	.1

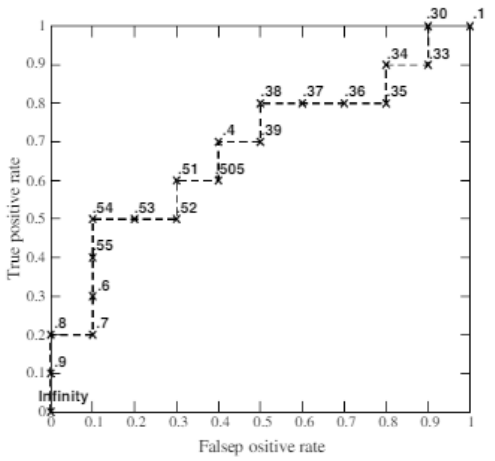

Figura 5.5: Ejemplo de curva ROC, extraido del trabajo de Fawcett [19], en el que se incluyen los resultados de cada muestra y su verdadera etiqueta,a partir de los cuales se obtienen los puntos de la gráfica. En cada punto se indica el valor del umbral mediante el que se han obtenido los ratios

CAPÍTULO 6
Resultados

En este capítulo se recopilan los resultados obtenidos en la tarea de detección de exoplanetas por cada una de las arquitecturas neuronales implementadas, así como el efecto que han tenido la aplicación de las técnicas de *data augmentation* implementadas.

6.1 Red LSTM

Combinación capas	Accuracy	Precision	Recall
1 capa, 8 unidades	67.40 %	56.17 %	49.16 %
1 capa, 16 unidades	67.96 %	58.78 %	41.28 %
1 capa, 32 unidades	68.77 %	58.80 %	48.69 %
1 capa, 64 unidades	67.95 %	57.22 %	48.85 %
1 capa, 128 unidades	68.90 %	65.04 %	32.32 %
1 capa, 256 unidades	69.06 %	58.80 %	51.40 %
2 capas, 8 unidades	67.34 %	56.00 %	49.72 %
2 capas, 16 unidades	68.66 %	60.47 %	41.19 %
2 capas, 32 unidades	68.64 %	60.84 %	35.99 %
2 capas, 64 unidades	69.22 %	61.00 %	48.69 %
2 capas, 128 unidades	69.17 %	57.07 %	63.29 %
2 capas, 256 unidades	68.83 %	60.68 %	41.94 %
3 capas, 8 unidades	67.32 %	57.94 %	38.24 %
3 capas, 16 unidades	67.49 %	57.47 %	42.50 %
3 capas, 32 unidades	68.00 %	57.45 %	48.01 %
3 capas, 64 unidades	68.75 %	60.21 %	42.78 %
3 capas, 128 unidades	69.71 %	61.95 %	44.46 %
3 capas, 256 unidades	65.82 %	54.36 %	40.57 %

Tabla 6.1: Tabla de resultados para la red recurrente LSTM sin aplicar *data augmentation*

Tras haber entrenado varias iteraciones de las distintas variaciones de nuestra red LSTM sobre el *data set* de entrenamiento completo, sin aplicar ninguna de las técnicas de *data augmentation*, obtenemos los resultados que podemos ver en Tabla 6.1. Cada fila de esta tabla muestra una posible combinación de capas LSTM y 3 métricas obtenidas como resultado de la mejor iteración de dicha combinación. Los resultados no varían excesivamente entre una red u otra ,pero podemos apreciar un patrón, cuanto mayor el tamaño de la salida de las redes LSTM, mejores resultados obtenemos. En términos de número de muestras positivas correctamente clasificadas, la red que ha obtenido mejores resultados está compuesta por 2 capas LSTM de 128 unidades cada una.

Sin embargo, a pesar de que este tipo de red es capaz de clasificar las muestras de test con un *accuracy* de aproximadamente el 70 % y un *precision* del 60 %, la mayoria de los resultados tienen en común un pésimo valor de *recall*, no superando el 50 % en casi la totalidad de las combinaciones. Estos resultados parecen indicar que nuestras redes LSTM no aprenden correctamente a clasificar las muestras positivas y que por lo tanto los resultados de casi un 70 % de *accuracy* se basan en clasificar la mayoría de muestras como negativas, siendo esta la clase mayoritaria ocupando un 61 % del *data set* total frente al 39 % de muestras positivas disponibles.

Como ya hemos dicho, los resultados de la Tabla 6.1 corresponden a los modelos entrenados sin la aplicación de las técnicas de *data augmentation* implementadas en nuestro *data generator*. Al realizar la misma experimentación con estas técnicas se ha encontrado que estas no tienen ningún efecto positivo sobre los resultados, o bien se obtienen valores prácticamente idénticos o impiden que el entrenamiento del modelo pueda avanzar lo suficiente, convergiendo a una peor solución.

Modelo	Precision	Recall	AUC
LSTM (nuestro)	57.07 %	63.29 %	0.753
LSTM (Hinners et al.)	52.0 %	-	-

Tabla 6.2: Tabla de resultados comparativa con nuestro modelo LSTM

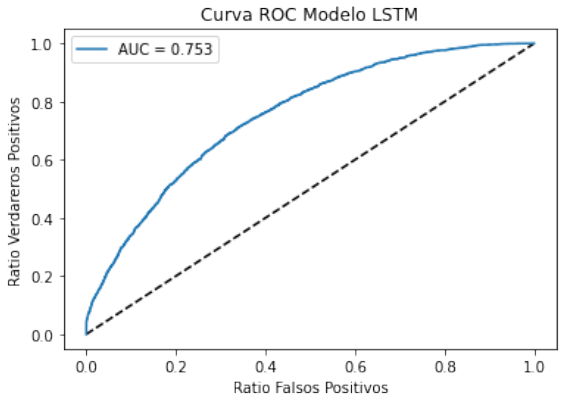

Figura 6.1: Curva ROC resultado de la clasificación con el modelo recurrente LSTM

Si nos centramos en la combinación de capas que mejor resultado de *recall* ha obtenido, es decir, que mayor cantidad de muestras positivas ha reconocido correctamente, podemos representar el resultado de su clasificación en la curva ROC de la Figura 6.1. La curva ROC obtenida posee un área con valor 0.753 y supera con creces el resultado que obtendríamos simplemente clasificando las muestras aleatoriamente en una de las dos clases, representado en la gráfica en forma de linea discontinua. En cuanto a la comparación con los resultados de otros estudios, en la Tabla 6.2 podemos observar como nuestro modelo ha logrado superar el trabajo de Hinners et al., cuya publicación ha representado nuestro punto de partida para el entrenamiento del modelo LSTM.

6.2 Red Convolucional 1D

En el caso de las redes convolucionales no se incluyen los resultados de diferentes configuraciones en las estructura de capas dado que la red base está basada en el modelo empleado en otros trabajos al que hemos aplicado distintas técnicas propias de las redes

neuronales, como capas *dropout* o control del *learning rate*; generando la red que puede encontrarse en el esquema de la Figura 5.2 en el capítulo anterior.

Modelo	Precision	Recall	AUC
Convolucional 1D (nuestro)	88.49 %	76.56 %	0.89
AstroNet (Shallue & Vanderburg)	93.00 %	95.00 %	0.98
Gradient Boost (Malik et al.)	82.00 %	96.00 %	0.94

Tabla 6.3: Tabla comparativa de resultados para la red convolucional 1D

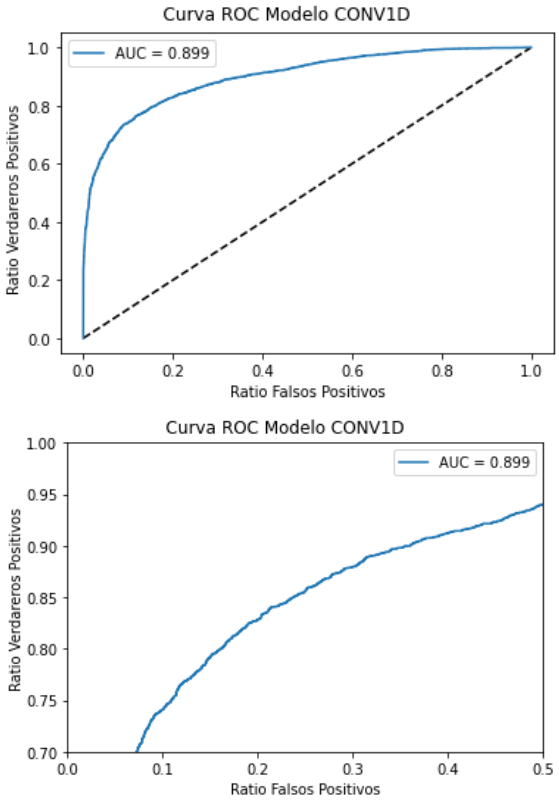

Figura 6.2: Curva ROC resultado de la clasificación con el modelo convolucional 1D y curva ROC ampliada

En este caso, la red convolucional 1D implementada es capaz de distinguir correctamente un 76.6 % de las muestras positivas, tal y como muestra el valor de *recall* obtenido, y llega a alcanzar un *precision* de más del 88 % en la clasificación general de todas las muestras. Estos resultados superan con creces a los obtenidos por la red LSTM. Si comparamos los resultados con los de otros trabajos similares (Tabla 6.3), veremos que nuestros resultados se sitúan entre los trabajos de Malik et al. y el actual estado del arte representado por el modelo de Shallue y Vanderburg. Los valores de *precision* obtenidos son competitivos en cuanto al estado del arte, sin embargo, el resultado de *recall* de nuestra red no es tan positivo. Existe una gran diferencia entre el *recall* del resto de trabajos y el nuestro, lo que parece indicar que nuestro modelo posee una predisposición a clasificar la mayoría de las muestras como negativas.

A pesar de sus diferencias con los resultados de la red LSTM, la red convolucional tampoco parece beneficiarse de las técnicas de *data augmentation*, siendo los valores de la tabla los correspondientes al entrenamiento del modelo sin aplicar dichas técnicas.

6.3 Red Convolucional 2D

Por último, presentamos los resultados del entrenamiento de la red convolucional 2D. En el trabajo original el que nos hemos basado para esta implementación, esta red era capaz de igualar los resultados de la red convolucional 1D, sin embargo estos resultados estaban basados en el uso de muestras simuladas. Tras entrenar nuestros modelos con muestras reales podemos apreciar un descenso en cada una de las métricas al comparar los resultados obtenidos con aquellos de la red 1D. El valor de *precision* obtenido por la red 2D es comparable al obtenido por la 1D. Por otro lado el valor de *recall* ha descendido aun más en este caso en comparación con el resto de trabajos que hemos explorado. De la misma forma, el área de la curva ROC también ha descendido.

Modelo	Precision	Recall	AUC
Convolucional 2D (nuestro)	87.5 %	70.36 %	0.86
AstroNet (Shallue & Vanderburg)	93.00 %	95.00 %	0.98
Gradient Boost (Malik et al.)	82.00 %	96.00 %	0.94

Tabla 6.4: Tabla comparativa de resultados para la red convolucional 2D

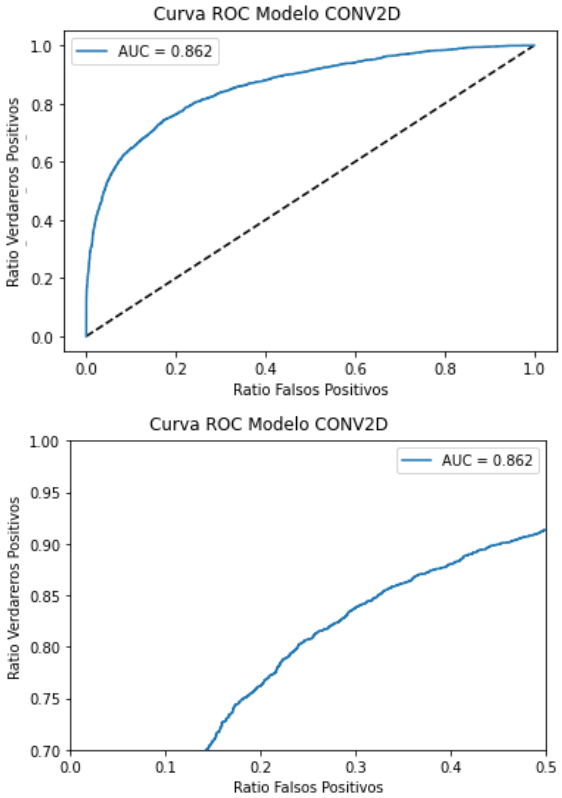

Figura 6.3: Curva ROC resultado de la clasificación con el modelo convolucional 2D y curva ROC ampliada

CAPÍTULO 7
Conclusiones

En este capítulo se recogen las conclusiones alcanzadas durante el desarrollo de este trabajo de final de máster, desde el punto de vista de los resultados obtenidos, pero también teniendo en cuenta los conocimientos adquiridos durante la fase de investigación del estado del arte, o los problemas a los que nos hemos enfrentado durante la implementación de los modelos.

Durante el desarrollo de este estudio se ha llevado a cabo, en primer lugar, una investigación del estado del arte en el ámbito de la aplicación de técnicas de *Machine Learning* a la búsqueda y detección de exoplanetas. Dicha investigación nos ha permitido partir desde una base sobre la que construir nuestros modelos y planear nuestros experimentos, al mismo tiempo que nos ha permitido disponer de unos resultados con los que comparar nuestros propios. Sin embargo, la exploración realizada no se ha limitado a trabajos y estudios similares, durante las primeras fases de este proyecto fue necesaria una búsqueda de posibles fuentes de datos con los que trabajar durante el resto del trabajo. Gracias a esta fase de exploración, hemos sido capaces de comprender el funcionamiento de distintas misiones astronómicas más allá del campo de la informática, adquiriendo conocimientos sobre los instrumentos y las técnicas empleadas en este tipo de misiones. Como requisito para elaborar el *data set* de muestras etiquetadas que finalmente hemos empleado para entrenar y comprobar la eficacia de cada modelo, hemos desarrollado un *pipeline*, realizando diversas tareas de limpieza y filtrado de datos, requiriendo no solo conocimientos propios del campo de la inteligencia artificial y haciendo uso de técnicas de tratamiento de datos.

En cuanto a la fase de experimentación con distintos modelos de *Machine Learning*, gracias a este trabajo hemos podido conocer como aplicar diferentes redes neuronales a un tipo de datos concreto, las series temporales univariadas. Las series temporales poseen una relación de continuidad entre los distintos valores que las conforman, la cual debe de tenerse en cuenta a la hora de seleccionar los modelos que se van a emplear para resolver el problema. Por ejemplo, en este trabajo hemos decidido emplear redes basadas en capas LSTM, las cuales son capaces de almacenar estados anteriores de la red y emplearlos en posteriores procesos de clasificación, lo que permite a la red establecer relaciones basadas en el orden en el que las series son analizadas. Si bien las redes convolucionales no se especializan en la detección de relaciones temporales en series, sí son útiles en la detección de patrones, como por ejemplo los descensos en la intensidad del flujo de las estrellas reflejados en las series temporales, por lo que al igual que las redes LSTM han resultado una elección acertada con la que abordar este problema. Las redes convolucionales, por su parte, representan el verdadero estado del arte para nuestro problema, alcanzado tasas de precisión superiores tanto en la literatura explorada como en nuestros experimentos. A diferencia de lo que seria lógico imaginar al comparar las estructuras de ambos tipos de redes, esta mejora en los resultados en las redes convolucionales no viene

acompañado de un aumento en el tiempo de computo. Pues a pesar de que las redes convolucionales requieren aproximadamente 10 veces más parámetros entrenables que las LSTM, son capaces de aplicar el mismo entrenamiento en un tiempo considerablemente inferior.

A primera vista, los resultados obtenidos en nuestros experimentos muestran datos positivos y alentadores en cuanto a la aplicación de modelos neuronales en problemas típicos de la astrofísica. En general, cada una de las 3 arquitecturas ha sido capaz de enfrentarse al problema de la clasificación de series temporales positivamente. Sin embargo, si analizamos las arquitecturas por separado, los resultados obtenidos por la red LSTM para las diferentes métricas empleadas generan dudas en cuanto a la validez de estos. En general, los resultados obtenidos por las diferentes configuraciones de nuestro modelo LSTM a penas superan el 50 % en la métrica *recall*, indicando que las redes solamente clasifican correctamente la mitad de las muestras positivas, lo que es equivalente a clasificarlas aleatoriamente al solo existir dos posibles clases. En consecuencia, los valores de *precision* en torno al 60 % serían consecuencia del desequilibrio de clases, pues al realizar una clasificación básicamente aleatoria, la red clasifica correctamente más muestras de la clase mayoritaria. A pesar de ello, la combinación de capas que mejor resultado a alcanzado demuestra que es posible mejorar estos resultados generales.

Las redes convolucionales han demostrado ser mucho más eficaces en la tarea detección de exoplanetas. Ambas arquitecturas han sido capaces de clasificar correctamente más del 80 % de las muestras disponibles y al menos el 70 % de las muestras etiquetadas como positivas. Las capas convoluciones han demostrado poder reconocer los patrones propios de los exoplanetas en los flujos de radiación registrados y, a diferencia de las capas LSTM, los resultados de las diferentes métricas generan mayor confianza en la eficacia de este método. La comparación con otros trabajos similares deja patente a su vez la calidad de los resultados que se han obtenido. El trabajo de Jiang et al. [6], sobre el cual hemos basado la estructura básica de nuestras redes convolucionales y la metodología para el uso de redes convolucionales 2D, defiende que es posible alcanzar unos valores de *accuracy* superiores al 99 % empleando dichas arquitecturas. Sin embargo, en el trabajo de estos autores puede estar cometiendose un error y los resultados no ser representativos de la eficacia real de las redes neuronales. Como muestras de entrenamiento de las redes de Jiang et al. se emplearon muestras simuladas; este hecho como tal no representa ningún error, pues puede ser útil para aumentar el número de muestras disponibles, al igual que mediante el uso de *data augmentation*. Sin embargo, su trabajo describe como se emplearon muestras simuladas tanto para entrenamiento, como para validación y test de las redes; el uso de muestras simuladas en la fase de test obviamente no permite la evaluación de la eficacia de la red sobre muestras reales. Adicionalmente, el uso de muestras simuladas en cada una de las fases del entrenamiento de las redes supone el riesgo de que los modelos aprendan a distinguir, no los atributos que caractericen las muestras con información de exoplanetas de aquellas que no, si no las características propias del sistema de simulación que se haya empleado para generar las muestras, de manera que sea completamente incapaz de reconocer e identificar muestras no generadas de esta manera. Aceptamos que las redes empleadas por Jiang et al. son capaces de resolver el problema de la detección de exoplanetas, pero creemos que los resultados que presentan podrían ser excesivamente optimistas.

En relación a otros trabajos, como los de Shalue y Vanderburg [4] o los de Malik et al.[5], los resultados de nuestros modelos convolucionales han demostrado ser competitivos en el actual estado del arte. El trabajo de Malik et al. emplea un modelo clasificador basado en *gradient boosting* y arboles de decisión para realizar la clasificación de muestras en dos clases. Sus resultados sobre los datos obtenidos por el satélite *Kepler* logran un valor de *precision* del 82 %, en comparación a nuestro propio resultado de entre un 88 %

y un 87 % para ambos modelos convolucionales implementados. Mientras que el *recall* obtenido por su trabajo alcanza hasta un 96 % , una cifra notablemente superior a la de nuestro modelo. Los resultados de Shallue y Vanderburg, considerados como el verdadero estado del arte por otras publicaciones, logran hasta un 92 % en la métrica *precision* y valores similares de *recall* a los del trabajo de Malik et al. Comparativamente, nuestros modelos son capaces de alcanzar resultados competitivos con estos trabajos, pero carecen de la fiabilidad de estos para identificar correctamente la mayor cantidad posible de muestras verdaderamente positivas.

Como conclusión final, creemos que se han alcanzado satisfactoriamente cada uno de los objetivos planteados para este proyecto. Se ha realizado una correcta investigación del estado del arte y las posibles fuentes de datos; se han desarrollado diferentes modelos neuronales y comparado sus resultados empleando métricas adecuadas; y, por último, los resultados conseguidos han demostrado ser relevantes en el actual estado del ámbito de la detección y clasificación de exoplanetas.

CAPÍTULO 8
Propuestas de trabajo futuro

En este capítulo final proponemos una posible ampliación a futuro, con el propósito de ampliar los resultados que aquí hemos presentado. Esta propuesta partiría de aplicar la misma metodología que hemos desarrollado para la construcción de nuestro *data set* a conjuntos de datos diferentes, por ejemplo aquellas series temporales de flujo registradas por otras misiones espaciales diferentes de *Kepler*. Actualmente, podemos acceder con la misma facilidad que a los datos recogidos por *Kepler*, a los datos de las misiones *K2* y *TESS*. Todos estos datos se encuentran en el *Mikulski Archive for Space Telescopes*, cada uno accesible a través de los distintos métodos de descarga que hemos presentado en el Capítulo 3. El principal problema que podría surgir al tratar con estos nuevos datos sería el proceso de etiquetado, necesario para el entrenamiento de nuestras redes. Al haber sido la primera misión lanzada, los datos de *Kepler* han sido revisados extensivamente y ha sido posible construir un completo catalogo de estos en función de la presencia de exoplanetas. Por otra parte, los datos de las misiones *K2* y *TESS* han dispuesto de menos tiempo para ser revisados y todavía no se han clasificado correctamente la totalidad de las muestras de estas misiones. A medida que pase el tiempo, estos nuevos datos deberían de seguir recibiendo revisiones hasta que finalmente pueda publicarse un catalogo de muestras etiquetadas, al igual que el disponible en el *Nasa Exoplanet Archive* para *Kepler*.

Otra posible ampliación partiría de revisar la implementación de las técnicas de *data augmentation* que hemos empleado. El hecho de que ninguna de estas técnicas haya sido capaz de aumentar los resultados en ninguno de los modelos resulta incoherente. Una correcta implementación debería lograr posponer el sobre-entrenamiento de los modelos, permitiéndoles alcanzar mejores resultados sobre los conjuntos de test y validación.

Bibliografía

[1] Wolszczan, A. and D. Frail. “A planetary system around the millisecond pulsar PSR1257 + 12.” Nature 355 (1992): 145-147.

[2] Jason Wei. “A Survey of Exoplanetary Detection Techniques“. arXiv preprint arXiv:1805.02771

[3] Deeg H.J., Alonso R. (2018) Transit Photometry as an Exoplanet Discovery Method. In: Deeg H., Belmonte J. Handbook of Exoplanets. Springer, Cham. https://doi.org/10.1007/978-3-319-30648-3_117-1

[4] C. J. Shallue, A. Vanderburg, Identifying Exoplanets with Deep Learning: A Five-planet Resonant Chain around Kepler-80 and an Eighth Planet around Kepler-90. Astron. J.

[5] Malik, Abhishek & Moster, Ben & Obermeier, Christian. (2020). Exoplanet Detection using Machine Learning.

[6] Jiang, Ing-Guey & Chintarungruangchai, Pattana. (2019). Detecting Exoplanet Transits through Machine-learning Techniques with Convolutional Neural Networks. Publications of the Astronomical Society of the Pacific. 131. 064502. 10.1088/1538-3873/ab13d3.

[7] Ian Goodfellow and Yoshua Bengio and Aaron Courville.(2016) "Deep Learning",MIT Press. Disponible en: http://www.deeplearningbook.org

[8] Space Telescope Science Institue’s Home Page. Consultado Consultado el 6 de Agosto de 2021 en https://www.stsci.edu/

[9] Barbara Mikulski Archive for Space Telescopes’ Home Page. Consultado el 6 de Agosto de 2021 en https://archive.stsci.edu/

[10] Hinners, T. A., Tat, K., & Thorp, R. (2018). "Machine learning techniques for stellar light curve classification". The Astronomical Journal,156,7

[11] Staudemeyer, Ralf & Morris, Eric. (2019). Ünderstanding LSTM – a tutorial into Long Short-Term Memory Recurrent Neural Networks".

[12] Yu, C., Li, K., Zhang, Y., Xiao, J., Cui, C., Tao, Y., Tang, S., Sun, C., & Bi, C. (2021). .A survey on machine learning based light curve analysis for variable astronomical sources". Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, e1425. https://doi.org/10.1002/widm.1425

[13] "The Kepler Science Data Processing Pipeline Source Code Road Map", publicado el 13 de Octubre de 2016. Consultado en https://ntrs.nasa.gov/api/citations/20170005719/downloads/20170005719.pdf.

[14] "MAST Kepler Archive Manual", publicado el 14 de Julio de 2020. Consultado en: https://archive.stsci.edu/missions-and-data/kepler

[15] "Kepler Data Processing Handbook", Capítulo 8, publicado el 20 de Enero de 2017. Consultado en: https://archive.stsci.edu/files/live/sites/mast/files/home/missions-and-data/kepler/_documents/KSCI-19081-002-KDPH.pdf

[16] Akeson, R. L., "The NASA Exoplanet Archive: Data and Tools for Exoplanet Research", Publications of the Astronomical Society of the Pacific, vol. 125, no. 930, p. 989, 2013. doi:10.1086/672273.

[17] Chollet, François and others, "Keras", 2015. Consultado en: https://keras.io

[18] Martín Abadi and others. "TensorFlow: Large-scale machine learning on heterogeneous systems", 2015. Software disponible en: tensorflow.org

[19] Fawcett, Tom. (2006). Introduction to ROC analysis. Pattern Recognition Letters. 27. 861-874. 10.1016/j.patrec.2005.10.010.

[20] Khondoker Nazmoon Nabi, Md Toki Tahmid, Abdur Rafi, Muhammad Ehsanul Kader, Md. Asif Haider. Forecasting COVID-19 cases: A comparative analysis between recurrent and convolutional neural networks. Results in Physics, Volume 24, 2021, 104137, ISSN 2211-3797. Disponible en: https://doi.org/10.1016/j.rinp.2021.104137