UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
ESCUELA DE POSTGRADO Y EDUCACIÓN CONTINUA
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

APRENDIZAJE AUTO-SUPERVISADO PARA LA DETECCIÓN DE CURVAS DE LUZ ANÓMALAS

TESIS PARA OPTAR AL GRADO DE
MAGÍSTER EN CIENCIA DE DATOS

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL ELÉCTRICO

DANIEL ANDRÉS CARMONA GONZÁLEZ

PROFESOR GUÍA:
PABLO ESTÉVEZ VALENCIA

MIEMBROS DE LA COMISIÓN:
IGNACIO REYES JAINAGA
FRANCISCO FÖRSTER BURON

Este trabajo ha sido parcialmente financiado por ANID, Iniciativa Científica Milenio, ICN12_009 y el proyecto Fondecyt 1220829.

SANTIAGO DE CHILE
2024

RESUMEN DE LA TESIS PARA OPTAR
AL GRADO DE MAGÍSTER EN CIENCIA
DE DATOS y MEMORIA PARA OPTAR AL
TÍTULO DE INGENIERO CIVIL ELÉCTRICO.
POR: DANIEL ANDRÉS CARMONA GONZÁLEZ
FECHA: 2024
PROF. GUÍA: PABLO ESTÉVEZ VALENCIA

Aprendizaje Auto-Supervisado para la Detección de Curvas de Luz Anómalas

Se propone un modelo basado en aprendizaje auto-supervisado para la detección de curvas de luz anómalas mediante la asignación de un puntaje de anomalía. En este enfoque, cada curva de luz se somete a una codificación que la convierte en un vector de longitud fija utilizando un Autoencoder. Este Autoencoder está compuesto por capas convolucionales y capas LSTM para realizar esta codificación de manera eficiente. Después de obtener la representación codificada o *embedding* de la curva, se agrega a este vector un conjunto de características que se calculan a partir de la banda de observación g de las curvas de luz. Adicionalmente se realizan experimentos al utilizar características computadas a partir de la banda de observación g y r de manera independiente, y en conjunto.

El puntaje de anomalía se determina en función de la cercanía al clúster más cercano en el espacio de representación. Estas representaciones se obtienen mediante un modelo Perceptrón Multicapa que ha sido entrenado utilizando técnicas de aprendizaje auto-supervisado contrastivo.

Este enfoque se aplica a datos del sondeo *Zwicky Transient Facility* (ZTF) procesados por ALeRCE, que incluye curvas de luz Periódicas (ZTF-PER), Estocásticas (ZTF-STO) y Transientes (ZTF-TRA).

Tanto el modelo propuesto como los modelos de referencia se evalúan utilizando las métricas AUCPR y AUROC. En ambos casos, la clase positiva se considera como la clase que se etiqueta como *outlier*. Dentro de los modelos de referencia que forman parte del estado del arte incluyen enfoques como One-Class SVM, Local Outlier Factor (LOF), TS-TCC y MCDSVDD. Estos modelos se utilizan como puntos de comparación para evaluar el rendimiento del modelo propuesto en la detección de objetos inusuales en los datos.

Los resultados obtenidos muestran que el enfoque propuesto supera a los métodos existentes en la detección de la mayoría de las clases de objetos astronómicos en los conjuntos de datos Transientes y Estocásticos. Sin embargo, obtiene resultados inferiores para la mayoría de las clases de objetos Periódicos con respecto a los modelos de referencia. La importancia de este trabajo radica en la capacidad del aprendizaje auto-supervisado para aprovechar al máximo los datos no etiquetados, que suelen ser mucho más abundantes que los datos etiquetados en muchos campos, incluida la astronomía, donde la recopilación de datos es constante y abarca un vasto espectro de información.

Para mi familia y amigos.

Agradecimientos

En primer lugar, deseo expresar mi profundo agradecimiento a mis padres, Juan Carmona y Marcela González, por el amor y cariño con el que me criaron, por las lecciones de vida que me transmitieron y por su apoyo inquebrantable a lo largo de mi trayecto académico y personal. A mis hermanos Rubén y Alejandro, quienes siempre estuvieron a mi lado, y a mi querida Catalina por darme ánimo y acompañarme en todo momento.

También quiero extender mi agradecimiento a mis amigos, con quienes compartí numerosas horas de estudio y momentos de distracción y diversión. Especialmente, agradezco a Consuelo Rojas, quien estuvo a mi lado durante la mayor parte de mi tiempo en la Universidad.

Mi reconocimiento se dirige al profesor Pablo Estévez por su valiosa orientación durante mi programa de magíster. Asimismo, quiero expresar mi gratitud a los miembros del laboratorio de inteligencia por brindarme críticas constructivas constantes en cada una de nuestras reuniones.

Por último, pero no menos importante, deseo agradecer al proyecto ANID, Iniciativa Científica Milenio ICN12009, y al Fondecyt Regular 1220829 por su apoyo financiero, que fue fundamental para llevar a cabo esta tesis.

Tabla de Contenido

Índice de Tablas

Índice de Ilustraciones

Capítulo 1

Introducción

1.1. Motivación

En la actualidad, la mayor parte de la investigación astronómica se lleva a cabo a partir del análisis de datos adquiridos de observatorios terrestres, los cuales generan una gran cantidad de datos cada noche. En el caso de observatorios ópticos, estos adquieren imágenes del cielo nocturno que pueden ser transformadas a series de tiempo de luminosidad (curvas de luz). Una característica de estas curvas de luz es que son muestreadas de forma irregular, son de largo variable y poseen más de una banda debido a que pueden ser obtenidos a partir de diferentes regiones del espectro electromagnético [4].

La automatización del procesamiento y análisis de grandes cantidades de datos ha sido objeto de estudio de diversos investigadores. Sin embargo, a la hora de procesar y analizar esta información recopilada surge el problema de detección de novedades o anomalías, es decir, la detección de objetos que se encuentran fuera de lo normal.

Debido al volumen de datos astronómicos generados cada noche, se cuenta con una gran cantidad de estos sin etiquetar. El proceso de etiquetado de la información requiere de mucho tiempo, por lo que surge la necesidad de aprovechar estas grandes cantidades de datos sin etiquetar para el entrenamiento de modelos de detección de anomalías. Es aquí donde aparece el aprendizaje auto-supervisado, el cual tiene como motivación, aprender representaciones útiles de los datos en un grupo sin etiquetar y luego ajustar las representaciones con una pequeña cantidad de datos etiquetados [5], lo cual se conoce como *fine-tuning*.

Dentro de los métodos de aprendizaje auto-supervisado más utilizados se encuentra el aprendizaje contrastivo. Para entender cómo funcionan los métodos de aprendizaje contrastivo, supongamos que tenemos una función f (representada por una red profunda), la cual a partir de una entrada x nos entrega las características $f(x)$ como salida de la red. El aprendizaje contrastivo establece que para cualquier par positivo x_1 y x_2 de datos, sus respectivas salidas $f(x_1)$ y $f(x_2)$ deben ser similares entre sí y para una entrada negativa x_3, $f(x_1)$ y $f(x_2)$ deben ser diferentes a $f(x_3)$ a partir de la métrica de similitud utilizada. Bajo esta lógica, se considera un par positivo, a datos pertenecientes a una misma clase, recortes distintos de una misma imagen o un mismo dato al cual se le aplicaron transformaciones distintas,

mientras que una entrada negativa corresponde a un dato perteneciente a una clase distinta, recorte de otra imagen, etc [6, 7].

De esta forma surgen técnicas de aprendizaje contrastivo auto-supervisado donde el enfoque se basa en la dualidad fundamental entre las vistas de tiempo y frecuencia de las señales de tiempo [2], o como en el método de Discriminación de instancias, donde se establece que dos versiones aumentadas de la misma imagen (par positivo) deben tener representaciones similares e invariantes (y representaciones diferentes para el par negativo) [8, 9].

1.2. Hipótesis

Se plantea la hipótesis de que un modelo de deep learning basado en aprendizaje auto-supervisado sería capaz de detectar curvas de luz atípicas o novedosas en astronomía de manera efectiva, superando los métodos existentes en la detección de *outliers* basada en curvas de luz. Se espera que este modelo tenga la capacidad de aprender representaciones significativas de los datos y descubrir patrones subyacentes sin depender de información previa sobre las clases a las que pertenecen.

1.3. Objetivos Generales

El objetivo general de esta tesis es desarrollar e implementar un modelo de deep learning basado en aprendizaje auto-supervisado para la detección de curvas de luz atípicas o novedosas.

El modelo propuesto se construye utilizando técnicas de aprendizaje auto-supervisado, donde se utilizan datos no etiquetados para entrenar al modelo en la extracción de características relevantes de las curvas de luz.

1.4. Objetivos Específicos

- Desarrollar e implementar un modelo de deep learning basado en aprendizaje auto-supervisado que sea capaz de asignar un puntaje de anomalía a una curva de luz. Esto implica diseñar la arquitectura del modelo, incluyendo la selección y configuración adecuada de las capas y algoritmos de aprendizaje.
- Evaluar el modelo implementado utilizando el dataset Zwicky Transient Facility (ZTF) de ALeRCE.
- Comparar el desempeño del algoritmo propuesto con el estado del arte en detección de outliers basado en curvas de luz, utilizando métricas de rendimiento adecuadas, como AUCPR y AUROC.

1.5. Metodología

1.5.1. Modelo Propuesto

El modelo propuesto para la detección de anomalías astronómicas corresponde a un modelo basado en aprendizaje contrastivo mediante aprendizaje auto-supervisado, el cual debe ser capaz de procesar series de tiempo multidimensionales, muestreadas de forma irregular y de largo variable dentro de un mismo conjunto de datos. Para esto, se propone utilizar una red neuronal (Autoencoder) compuesta de capas convolucionales y LSTM, que codifique las curvas de luz en vectores de largo fijo.

1.5.2. Comparación con Modelos de Referencia

Para comparar el rendimiento del modelo propuesto se propone utilizar 3 modelos basados en características (Isolation Forest, One-class Support Vector Machine, Local Outlier Factor), y al menos 2 modelos basados en redes neuronales, donde ambos modelos corresponden a modelos entrenados mediante aprendizaje auto-supervisado, específicamente SimCLR adaptado para series de tiempo [9] y TF-C [2] ó TS-TCC [3].

1.5.3. Criterios de Evaluación

Para evaluar los modelos base y propuesto, se propone utilizar como criterio de evaluación tanto el área bajo la curva de precisión y exhaustividad (AUCPR, por sus siglas en inglés, Area Under Precision Recall Curve) como el área bajo la curva ROC (AUROC, por sus siglas en inglés, Area Under Receiver Operating Characteristic Curve) [2][10]. Estas métricas de rendimiento son especialmente útiles cuando se trabaja con conjuntos de datos desequilibrados y se centra en la detección de ejemplos positivos o anomalías, como en el caso de este trabajo.

1.6. Estructura de la Tesis

La estructura propuesta para la tesis consta de cinco capítulos, donde se describe el desarrollo del trabajo realizado. A continuación se detalla el contenido de cada capítulo:

Capítulo 1: Introducción

En este capítulo se presenta una introducción general al tema de investigación, se plantea la problemática que se aborda en la tesis y se exponen los objetivos y las hipótesis planteadas. Además, se brinda una visión general de la importancia del problema y se destacan las contribuciones de la investigación.

Capítulo 2: Marco teórico

El segundo capítulo se centra en proporcionar el marco teórico necesario para comprender el trabajo realizado. Se introducen los términos astronómicos relevantes relacionados con el tema de la tesis y se explican los conceptos y teorías fundamentales en los que se basa el estudio. También se presentan los algoritmos y técnicas de redes neuronales utilizados en el desarrollo de la investigación.

Capítulo 3: Metodología

En este capítulo se describe detalladamente la metodología seguida para llevar a cabo el trabajo de investigación. Se explica el proceso de preprocesamiento de los datos astronómicos utilizados, incluyendo las técnicas de limpieza y normalización. Además, se describe la arquitectura de los modelos de aprendizaje auto-supervisado utilizados, detallando los componentes y parámetros relevantes.

Capítulo 4: Resultados y análisis

En este capítulo se presentan los resultados obtenidos a través de la aplicación de la metodología descrita en el capítulo anterior. Se analiza la efectividad del modelo en la detección de eventos astronómicos atípicos.

Capítulo 5: Conclusiones y trabajo futuro

En el último capítulo se realizan las conclusiones generales del trabajo de tesis, contrastando los resultados obtenidos con las hipótesis y objetivos planteados inicialmente. Se destacan las contribuciones específicas de la investigación y se discuten las implicaciones de los hallazgos. Además, se sugieren áreas para futuras investigaciones relacionadas con el tema.

Capítulo 2

Marco Teórico

2.1. Astronomía

2.1.1. Curvas de Luz

Las curvas de luz son representaciones gráficas que muestran cómo varía el brillo de un objeto astronómico durante un período de tiempo [11]. Estas curvas suelen utilizarse para estudiar objetos como estrellas, galaxias, supernovas, quásares y otros fenómenos astronómicos.

En una curva de luz típica, el eje vertical (eje y) representa la magnitud de la luz recibida del objeto astronómico. La magnitud es una medida del brillo aparente del objeto astronómico, donde valores más pequeños indican un brillo mayor. El eje horizontal (eje x) representa el tiempo, mostrando cómo varía el brillo del objeto en función del tiempo. La Figura 2.1 muestra un ejemplo de una curva de luz periódica, utilizando la Fecha Juliana Modificada (MJD) como unidad de medida temporal.

Es importante tener en cuenta que las curvas de luz pueden ser específicas de una banda de frecuencia en particular. En astronomía, se utilizan diferentes bandas de frecuencia o rangos de longitud de onda para estudiar objetos astronómicos en distintas regiones del espectro electromagnético. Algunos ejemplos comunes de bandas de frecuencia incluyen la luz visible, el infrarrojo, el ultravioleta, los rayos X y los rayos gamma. Dependiendo del objeto estudiado y los objetivos de la investigación, se puede elegir una banda de frecuencia específica para construir la curva de luz.

Al analizar una curva de luz, los astrónomos pueden observar y estudiar patrones y características particulares, como variaciones periódicas en el brillo que pueden indicar la presencia de un sistema binario, una estrella pulsante u otro fenómeno cíclico. También pueden identificar eventos transitorios, como explosiones estelares o cambios bruscos en el brillo que pueden ser indicativos de procesos astrofísicos de interés.

Las curvas de luz son herramientas fundamentales en la astronomía para estudiar la variabilidad temporal del brillo de los objetos astronómicos y comprender mejor los fenómenos que ocurren en el universo.

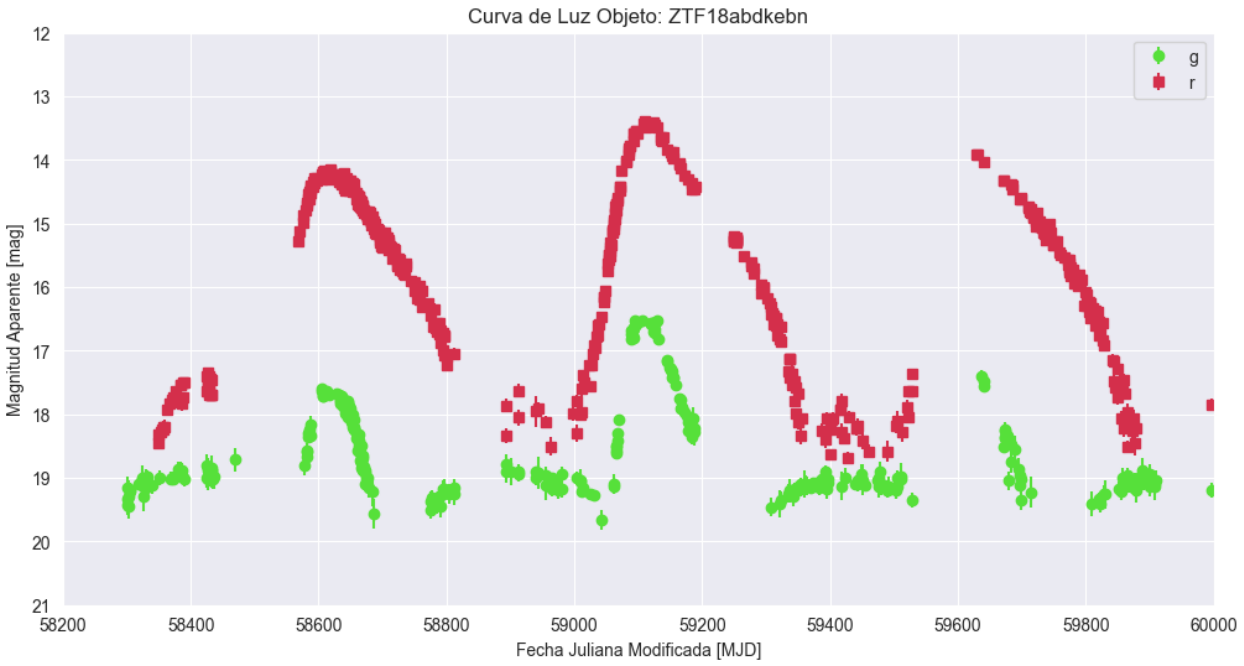


Figura 2.1: Ejemplo de curva de luz periódica, correspondiente al objeto astronómico *ZTF18abdkebn*, para las bandas g y r.

2.1.2. Luminosidad y Brillo

Se define la luminosidad (electromagnética) L como el total de energía que sale de la superficie de la fuente por unidad de tiempo en forma de fotones. La energía por unidad de tiempo se llama potencia, por lo que podemos medir L en las unidades físicas de potencia (unidades SI): joules por segundo o watts [12].

Otra medida de cantidad relacionada a la luminosidad, es el brillo aparente, denotado por F, la cual es mucho más simple de calcular. Esta métrica permite comparar la cantidad de energía que recibimos de diferentes objetos astronómicos, teniendo en cuenta la distancia y el área de detección. El flujo puede expresarse en unidades como joules por segundo por metro cuadrado $[\frac{J}{s \cdot m^2}]$. Matemáticamente, se expresa como:

$$F = \frac{E}{t \cdot A}\,, \tag{2.1}$$

donde F es el brillo aparente o *flujo* (también conocido como *densidad de flujo*), E es la energía total recibida desde la fuente de origen, t es el tiempo durante el cual se recibe la energía y A es el área a través de la cual se recibe la energía.

Sin embargo, las mediciones de flujo varían en órdenes de magnitud logarítmicos. De esta manera una estrella de primera magnitud es alrededor de 2.5 veces más brillante que una

estrella de segunda magnitud que, a su vez, es 2.5 veces más brillante que una estrella de tercera magnitud y así sucesivamente. De esta manera, una diferencia de cinco magnitudes corresponde a una relación de $2.5^5 = 100$ en brillo [13], por lo que la magnitud se define por la relación:

$$m = -2.5 \cdot \log_{10}(F) + K \,, \tag{2.2}$$

donde m es la magnitud aparente, F corresponde al brillo del objeto y la constante K se refiere a una constante escogida para que las medidas modernas coincidan, en cierta medida, con los catálogos más antiguos. Además, es posible determinar la diferencia de magnitud entre dos objetos mediante la siguiente ecuación:

$$m_1 - m_2 = -2.5 \cdot \log_{10}\left(\frac{F_1}{F_2}\right) \,, \tag{2.3}$$

donde m_1 y m_2 corresponden a las magnitudes de dos objetos astronómicos y F_1 y F_2 son flujos o brillos de los objetos correspondientes.

Cabe destacar que la magnitud aparente de un objeto astronómico también depende de la banda o intervalo de frecuencia que se utiliza para realizar la medición. De esta manera la magnitud m en la banda espectral x, se define:

$$m_x = -2.5 \cdot \log_{10}\left(\frac{F_x}{F_{x,0}}\right) \,, \tag{2.4}$$

donde F_x es la irradiancia observada usando el filtro espectral x, y $F_{x,0}$ es el flujo de referencia (punto cero) para ese filtro fotométrico.

2.1.3. Medición de Brillo a Partir de Imágenes

Para estimar el brillo o flujo de un objeto astronómico se utilizan principalmente dos tipos de mediciones: *fotometría* y *espectroscopía*.

La fotometría consiste en medir la cantidad total de luz (flujo) recibida de un objeto utilizando instrumentos especializados llamados fotómetros. Los fotómetros pueden estar equipados con filtros para medir la luz en bandas de longitud de onda específicas o pueden ser de banda ancha para medir la luz total en un amplio rango de longitudes de onda. Por otro lado, la espectroscopía implica la medición de la distribución de la intensidad de la luz en función de la longitud de onda. La espectroscopía proporciona información valiosa sobre la composición química, la temperatura y el movimiento de los objetos astronómicos (mediante el efecto Doppler)[14]. Además, las líneas espectrales y las características de absorción en el espectro se utilizan para identificar elementos y moléculas presentes en estrellas, galaxias y otros objetos astronómicos.

2.2. Machine Learning

El *Aprendizaje de Máquinas*, conocido en inglés como *Machine Learning* (ML) es un campo de estudio de la inteligencia artificial dedicado a la comprensión y creación de métodos que permiten que las máquinas aprendan, es decir, métodos que aprovechan los datos para mejorar el rendimiento de la computadora en algún conjunto de tareas. Para esto el ML utiliza una variedad de algoritmos que aprenden iterativamente de los datos para mejorar, describir los datos, encontrar patrones y predecir resultados, sin recurrir a una programación explícita.

Existen distintos tipos de entrenamiento de modelos de ML, dentro de los cuales se encuentran principalmente el aprendizaje supervisado, semi-supervisado, no supervisado, y aprendizaje auto-supervisado.

2.2.1. Aprendizaje Supervisado

Aprendizaje supervisado es un tipo de aprendizaje caracterizado por utilizar datos correctamente etiquetados durante el entrenamiento del modelo [15]. El objetivo de este tipo de algoritmo es aprender una función que pueda mapear correctamente un vector de características (entrada) a las etiquetas de salida, para posteriormente poder realizar predicciones o clasificaciones a partir de otro conjunto de datos [16].

Durante el entrenamiento, el modelo ajusta sus parámetros en base a los datos de entrenamiento minimizando una función de costos (o pérdida). La función de costos mide la discrepancia entre las predicciones del modelo y los valores reales de los datos de entrenamiento, por lo que la minimización de esta función implica ajustar los parámetros del modelo de manera que las predicciones se acerquen lo más posible a los valores reales (etiquetas).

En el aprendizaje supervisado, existen varios métodos y algoritmos ampliamente utilizados, algunos de los cuales incluyen: ciertos tipos de Redes Neuronales (*Neural Networks*), Máquinas de Vectores de Soporte (*Support Vector Machines, SVM*), K-Vecinos Más Cercanos (*K-Nearest Neighbors*).

2.2.2. Aprendizaje No-Supervisado

Este tipo de aprendizaje se caracteriza por utilizar conjuntos de datos que contienen muchas características, con la finalidad de que el modelo aprenda propiedades útiles como patrones o estructuras intrínsecas de los datos por sí mismo.

A diferencia del aprendizaje supervisado, el aprendizaje no supervisado funciona solamente con los datos que son ingresados al modelo, es decir, no se proporcionan etiquetas o variables objetivo, por lo que no se realizan correcciones durante el entrenamiento basados en la etiqueta correspondiente a los datos de entrada.

Existen diversos tipos de entrenamiento no supervisado, sin embargo, los principales métodos utilizados para este tipo de entrenamiento, corresponden a métodos basados en el agrupamiento de datos (*Clustering*), análisis de componentes principales (*PCA*), *Autoencoders* y métodos de estimación de densidad que implican encontrar la distribución de los datos [17].

2.2.3. Aprendizaje Semi-Supervisado

El aprendizaje semi-supervisado es un enfoque intermedio entre el caso supervisado y el no supervisado, ya que se utilizan tanto conjuntos de datos que se encuentran correctamente etiquetados, como ejemplos no etiquetados.

La idea detrás del aprendizaje semi-supervisado es aprovechar la información adicional proporcionada por los datos no etiquetados para mejorar el rendimiento del modelo. Esto, debido a que obtener ejemplos etiquetados puede ser costoso o requerir mucho tiempo y esfuerzo, mientras que los ejemplos no etiquetados pueden estar disponibles en grandes cantidades.

Hacer un uso eficaz de los datos no etiquetados puede requerir el uso de métodos no supervisados, como el agrupamiento y la estimación de la densidad. Una vez descubiertos los grupos o patrones que describen los datos, se pueden utilizar métodos de aprendizaje supervisado o ideas de este tipo de aprendizaje para etiquetar los ejemplos no etiquetados o aplicar etiquetas a las representaciones obtenidas que no se encuentran etiquetadas, las cuales son posteriormente utilizadas para la predicción [17].

2.2.4. Aprendizaje Auto-Supervisado

El aprendizaje auto-supervisado (Self-Supervised Learning, SSL) es un enfoque innovador en el campo del aprendizaje automático que se basa en entrenar modelos utilizando datos sin procesar y sin etiquetas externas [18]. A diferencia de otros métodos de aprendizaje, en el SSL, el modelo no depende de etiquetas previamente asignadas por humanos para aprender patrones significativos en los datos. En cambio, durante el proceso de entrenamiento, el modelo se encarga de crear internamente un conjunto de tareas auxiliares que le permiten extraer información útil de los datos de entrada.

Una característica fundamental del SSL es que transforma el problema inicialmente no supervisado en un problema supervisado desde la perspectiva del modelo [19]. Esto se logra, por ejemplo, al enseñar al modelo a predecir una parte de los datos de entrada a partir de otra parte de esos mismos datos, creando así etiquetas implícitas.

Ejemplos de Modelos en Aprendizaje Auto-Supervisado:

- BERT (Bidirectional Encoder Representations from Transformers): BERT es un modelo de procesamiento de lenguaje natural (NLP) que utiliza el aprendizaje auto-supervisado [20]. Se entrena para predecir palabras enmascaradas en oraciones. Por ejemplo, en la oración "El gato se subió al ___", BERT debe predecir la palabra faltante "árbol". Esto le permite aprender representaciones contextuales de palabras y frases.
- SimCLR (Contrastive Learning for Unsupervised Representation Learning): SimCLR es un modelo de visión por computadora que emplea el SSL [9]. A través de la comparación de pares de imágenes, SimCLR aprende a maximizar la similitud entre imágenes con la misma clase y minimizarla entre imágenes de clases diferentes [21]. Esto resulta en representaciones eficaces y generalizables de imágenes.

Un ejemplo de funcionamiento puede ser el caso en que se trabaja con datos de imágenes, y se desea entrenar un modelo SSL. En vez de de utilizar las etiquetas de clases, se toma cada imagen, y se aplica transformaciones de forma aleatoria, las cuales pueden ser recortes, rotaciones, ajustes de color y brillo, para crear varias versiones de una misma imagen. Luego, se entrena el modelo para predecir cual es la imagen original a partir de estas imágenes transformadas.

Durante el entrenamiento, el modelo ajusta sus pesos para realizar estas predicciones internas. A medida que el entrenamiento avanza, el modelo aprende a capturar características relevantes y discriminativas en las imágenes, lo que le permite comprender mejor la estructura de los datos. Una vez entrenado, el modelo puede ser utilizado para tareas de clasificación de imágenes u otras tareas relacionadas con la visión por computadora, incluso si no se dispone de etiquetas de clase. Esto demuestra cómo el aprendizaje auto-supervisado puede generar representaciones útiles y mejorar el rendimiento en tareas específicas.

2.3. Redes Neuronales Artificiales

Las redes neuronales o redes neuronales artificiales (ANNs del inglés *artificial neural networks*) son modelos matemáticos de aprendizaje automático inspirados en el cerebro humano. Estos modelos imitan la forma en que las neuronas biológicas transmiten señales entre sí y son fundamentales en el campo del aprendizaje profundo.

Estas redes crean un sistema adaptable que las computadoras utilizan para aprender de sus errores y mejorar continuamente. De esta forma, las redes neuronales artificiales intentan resolver problemas complejos, como la realización de resúmenes de documentos o el reconocimiento de rostros, con una mayor precisión.

Si bien las redes neuronales artificiales (ANNs) pueden presentar una gran variedad de estructuras y funcionamientos, una representación básica de estos modelos incluye capas de nodos. Estas capas constan de una capa de entrada, una o más capas ocultas y una capa de salida. Cada nodo, o neurona artificial, está interconectado con otros y está caracterizado

por pesos y un umbral. Si la salida de una neurona individual supera el umbral especificado, se activa y transmite datos a la siguiente capa de la red. De lo contrario, no se transmiten datos a la capa siguiente [22].

2.3.1. Perceptrón

El perceptrón fue presentado por Frank Rosenblatt en 1958 ([23]), como un modelo capaz de aprender y realizar tareas de clasificación binaria. Este modelo matemático se basó en el concepto de una unidad computacional simple, que toma una o más entradas y produce una sola salida, modelada según la estructura y función de una neurona en el cerebro, razón por la cual se considera como una unidad de red neuronal o neurona artificial.

El modelo cuenta con cuatro partes principales que incluyen *valores de entrada* en forma de un vector $x = (x_1, \ldots, x_N)^T$ con N valores. Un *vector de pesos* $w = (w_1, \ldots, w_N)$ y adicionalmente *sesgo* b. Luego, se multiplican cada uno de los valores del vector de entrada por su peso correspondiente, y se calcula la *suma ponderada* como se muestra en la ecuación 2.5. Por último, al resultado de la suma neta se le aplica una *función de activación* que entrega una salida binaria como se muestra a continuación:

$$\Sigma w_i \cdot x_i = w_1 \cdot x_1 + w_2 \cdot x_2 + \ldots + w_N \cdot x_N \,, \tag{2.5}$$

$$y = f(\Sigma w_i \cdot x_i + b) \,. \tag{2.6}$$

2.3.2. Perceptrón Multicapa

De la sección anterior, se sabe que el Perceptron cuenta con una capa de entrada y una de salida, donde solamente en la capa de salida se realizan cálculos. A diferencia del modelo perceptron, una red neuronal multicapa contiene múltiples capas donde se realizan cálculos. Además, cuenta con capas que se encuentran entre la entrada y salida del modelo, las cuales se llaman capas ocultas (*hidden layers*) debido a que los cálculos realizados en estas capas no se encuentran visibles al usuario.

De esta manera se define al modelo Perceptron Multicapa (MLP [24]) como una arquitectura basada en una colección de capas *fully-connected*, donde la salida de una capa se encuentra conectada a los nodos de la siguiente capa. De esta forma, para un vector x que representa el vector de entrada al modelo, se tiene que la salida y se expresa como se muestra a continuación:

$$y = f_{\theta_1} \circ \ldots \circ f_{\theta_N}(x) \,, \tag{2.7}$$

donde $f_{\theta_1}, \ldots, f_{\theta_N}$ corresponden a las N diferentes capas totalmente conectadas utilizadas para definir el modelo MLP.

2.3.3. Redes Neuronales Convolucionales

Las redes neuronales convolucionales (CNN del inglés *Convolutional Neural Network*) son un tipo de ANN que procesa sus capas imitando al córtex visual del cerebro humano. Este tipo de red es una variación del modelo perceptrón multicapa, diseñado para ser aplicado en matrices bidimensionales, razón por la cual son muy efectivas para tareas de visión artificial, como la clasificación y segmentación de imágenes, entre otras aplicaciones. Si bien, inicialmente fueron utilizadas para clasificar datos de 2 dimensiones, estos modelos han sido generalizados para clasificar datos de 1 dimensión, 2 o n dimensiones.

Esta difícil tarea se aprende de forma supervisada mediante una red neuronal que sigue un conjunto de restricciones estructurales específicas:

- Extracción de Características: cada neurona en la red recibe entradas de un campo receptivo local en la capa anterior. Esto significa que cada neurona se especializa en extraer características locales de los datos de entrada. Una vez que se extrae una característica, su ubicación exacta se vuelve menos importante, siempre y cuando su posición relativa con respecto a otras características se conserve aproximadamente [1].
- Mapeo de Características: cada capa en la red está compuesta por varios mapas de características, y cada mapa de características tiene la forma de un plano en el cual las neuronas individuales comparten el mismo conjunto de pesos sinápticos. Esta restricción estructural tiene dos efectos beneficiosos: primero, impone invariancia a ciertos cambios al forzar la operación de un mapa de características mediante la convolución con un núcleo de tamaño pequeño seguido de una función sigmoide. En segundo lugar, reduce la cantidad de parámetros libres al utilizar pesos compartidos entre las neuronas.
- Submuestreo: después de cada capa convolucional, hay una capa que realiza submuestreo y promedio local. Esto disminuye la resolución del mapa de características, lo que a su vez reduce la sensibilidad de la salida del mapa de características a cambios y otras formas de distorsión en los datos.

La Figura 2.2 muestra la arquitectura de una red convolucional compuesta de una capa de entrada, cuatro capas ocultas, y una capa de salida. Esta red esta diseñada para procesar imágenes, como por ejemplo, el reconocimiento de caracteres de escritura a mano.

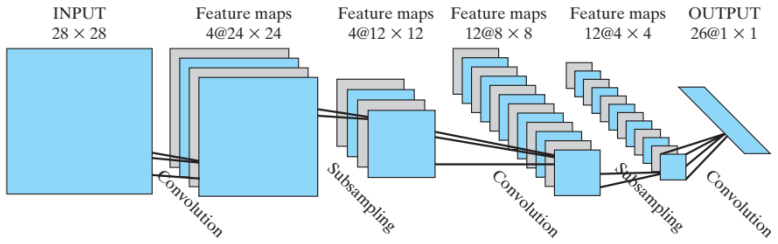


Figura 2.2: Arquitectura de una red neuronal convolucional, enfocada al reconocimiento de objetos en imágenes. Tomado de Neural Networks and Learning Machines, por Simon Haykin [1].

2.3.4. Redes Neuronales Recurrentes

Una red neuronal recurrente (RNN del inglés *Recurrent Neural Network*) es un tipo de red neuronal artificial que utiliza datos secuenciales como oraciones de texto (previamente codificadas) o series de tiempo.

En redes neuronales regulares, los datos pasan a través de una serie de capas. Para una sola observación, la salida de cada capa es la representación de la red neuronal en esa capa puntual. De esta forma, después de la primera capa, la representación de los datos consta de entidades que son combinaciones de las entidades originales, la siguiente capa, consiste en combinaciones de las representaciones previas y así sucesivamente para las capas posteriores de la red, conteniendo muchas representaciones de la observación original. El funcionamiento base de la RNNs es similar al proceso de una red regular, con la inclusión de representaciones pasadas junto al siguiente conjunto de observaciones [25]. Esto ultimo se puede definir como la siguiente ecuación:

$$h_t = f(h_{(t-1)}, x_t)\,, \tag{2.8}$$

$$y_t = g(h_t)\,, \tag{2.9}$$

donde h_{t-1} representa a un vector del estado previo, x_t representa al vector u observación de entrada y la función f que sirve de función de transición que utiliza ambos vectores. Adicionalmente, la función g_t es utilizada para aprender las probabilidades de salida provenientes del estado oculto actual h_t [26].

La Figura 2.3 corresponde al esquema de una red neuronal recurrente básica, y su correspondiente representación en capas de tiempo. La representación en capas de tiempo de una RNN se muestra como una serie de bloques conectados horizontalmente, donde cada uno de estos bloques representa una copia de la red en un paso de tiempo diferente. Esto ilustra cómo la RNN procesa y retiene información a lo largo de una secuencia, lo que es fundamental para muchas aplicaciones en las que se trabaja con datos secuenciales.

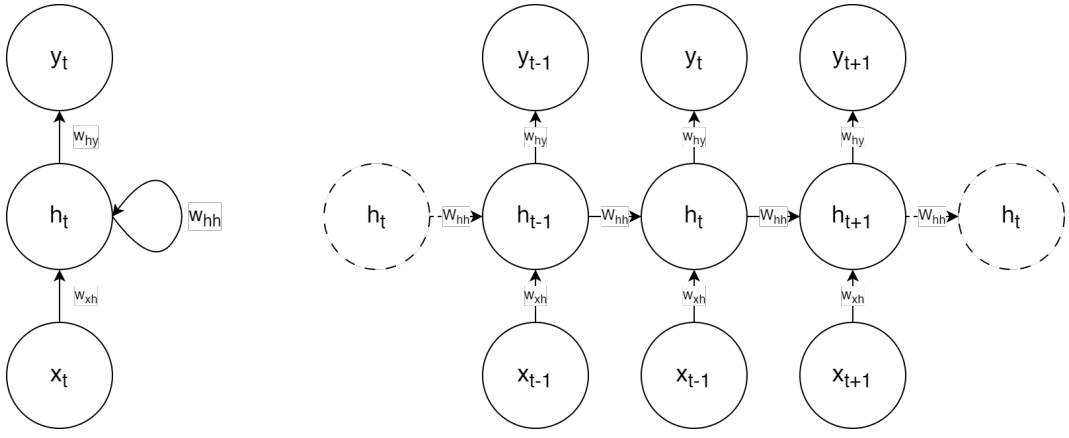


Figura 2.3: Esquema de una red neuronal recurrente básica y su representación en capas de tiempo, donde cada uno de los bloques conectados horizontalmente representa una copia de la red en un paso de tiempo diferente.

Modelo Long Short Term Memory (LSTM)

Uno de los problemas presentes en las *RNN* simples es que presentan memoria de corto plazo, debido al desvanecimiento de gradiente a medida que transcurren las épocas en el entrenamiento del modelo. Para solucionar este problema, se crearon redes neuronales que permiten que la información persista en el tiempo, lo cual es posible haciendo que los pesos dentro del bucle sea controlado por otra unidad en la capa oculta [26].

Las redes recurrentes LSTM (Long Short Term Memory), fueron diseñadas por Hochreiter y Schmidhuber [27] especialmente con el objetivo para manejar datos secuenciales, como series de tiempo, voz y texto. Este tipo de red neuronal es capaz de aprender dependencias de los datos a largo plazo, debido a la inclusión de celdas de memoria, que pueden contener información por varios periodos de tiempo o épocas de entrenamiento. Estas celdas de memoria son controladas por los siguientes componentes:

- *Estado Oculto (Hidden State)* estado oculto de tiempo anterior h_{t-1}. Este estado se combina con la entrada de tiempo actual x_t antes de que las copias de estos pasen a través de varias puertas.
- *Puerta de Olvido (Forget Gate)*: esta puerta controla qué información debe olvidarse. Dado que la función sigmoide oscila entre 0 y 1, establece qué valores en el estado de la celda deben ser descartados (se multiplica por 0), recordar (se multiplica por 1) o debe recordarse parcialmente (multiplicarse por algún valor entre 0 y 1) [28].
- *Puerta de Entrada (Input Gate)*: La puerta de entrada ayuda a identificar elementos importantes que deben agregarse al estado de la celda. Para esto, se multiplica los resultados de la entrada con el estado de la celda, y solo la información que la puerta considera importante se añaden al estado actual de la celda.
- *Puerta de Salida (Output Gate)*: esta puerta decide qué información en la celda de memoria se debe utilizar como salida.
- *Actualizar el Estado de Celda*: se multiplica el estado de la celda anterior c_{t_1} con los resultados de la puerta de olvido. Luego se agrega la información nueva para obtener el estado de celda actual c_t.
- *Actualizar Estado Oculto*: para actualizar el estado oculto, se pasa el ultimo estado de celda c_t a través de la función de activación *tanh* y se multiplica por los resultados de la puerta de salida.

Cada celda en una red neuronal LSTM (Long Short-Term Memory) tiene una estructura similar a la de una red recurrente estándar [26]. Sin embargo, se distingue por la presencia de parámetros y un mecanismo de puertas que regula el flujo de información. La unidad de estado clave en una LSTM es denotada como c_t, que incorpora un bucle de retroalimentación lineal autoregulable, controlado por la puerta de olvido (f_t^i). Esta puerta de olvido asigna valores entre 0 y 1 mediante una función sigmoide, como se muestra en la siguiente ecuación:

$$f_t^i = \sigma\left(b_f^i + \sum_j U_f^{i,j} x_t^j + \sum_j W_f^{i,j} h_{t-1}^j\right), \tag{2.10}$$

donde x_t representa el vector de entrada actual, h_t el vector de estado oculto actual (que incluye las salidas de todas las celdas LSTM), y b_f, U_f, W_f denotan, respectivamente, los sesgos, los pesos de entrada y los pesos recurrentes asociados a la puerta de olvido. De esta manera el estado interno de la celda LSTM se actualiza de la siguiente manera:

$$c_t^i = f_t^i c_{t-1}^i + g_t^i \sigma \left(b^i + \sum_j U^{i,j} x_t^j + \sum_j W^{i,j} h_{t-1}^j \right) , \tag{2.11}$$

donde b, U, W denotan, respectivamente, los sesgos, pesos de entrada y los pesos recurrentes de la celda LSTM. La puerta de entrada externa g_t^i se computa de forma análoga al caso de la puerta de olvido, pero con sus propios parámetros:

$$g_t^i = \sigma \left(b_g^i + \sum_j U_g^{i,j} x_t^j + \sum_j W_g^{i,j} h_{t-1}^j \right) . \tag{2.12}$$

La salida h(t) de la celda LSTM también puede ser controlada mediante la puerta de salida q_t^i, que también utiliza una unidad sigmoide para la regulación, como se muestra en la siguiente expresión:

$$h_t^i = \tanh \left(c_t^i \right) q_t^i , \tag{2.13}$$

$$q_t^i = \sigma \left(b_g^i + \sum_j U_o^{i,j} x_t^j + \sum_j W_o^{i,j} h_{t-1}^j \right) , \tag{2.14}$$

donde b_o, U_o, W_o denotan sus respectivos sesgos, pesos y pesos recurrentes. El procedimiento antes mencionado, se encuentra resumido en la Figura 2.4.

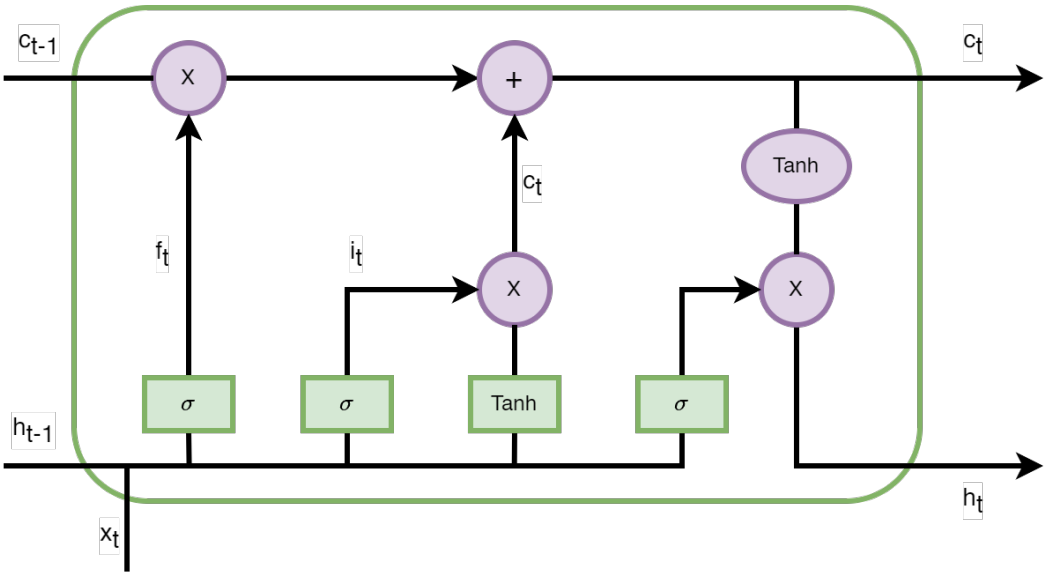


Figura 2.4: Esquema de una celda de una red neuronal recurrente LSTM.

2.3.5. Autoencoder

Un codificador automático o *Autoencoder* es una red neuronal entrenada de manera no supervisada para intentar replicar la entrada en la salida, habiendo pasado por un cuello de botella. Para esto, el modelo cuenta con capas ocultas denotadas como h_i, que describen una codificación utilizada para representar la información de entrada [26].

Esta codificación se logra a través de dos partes principales: una primera parte que codifica la información (Encoder) que genera $h = f(x)$ a partir de la entrada x, y una segunda función decodificadora (Decoder) que produce una reconstrucción a partir de la información codificada en la primera parte [26], como se muestra en la ecuación siguiente:

$$\hat{x} = g(h) \, . \tag{2.15}$$

Si bien el entrenamiento consiste en realizar una copia de los datos de entrada x, el propósito principal de entrenar un autoencoder es encontrar propiedades o características útiles que representen los datos. Esto se logra minimizando un función de costo L entre el vector de datos original y su versión reconstruida por el decodificador.

La función de costos que penaliza la reconstrucción $g(f(x))$ obtenida por el modelo por ser disimilar a la entrada x. Un ejemplo de función de costo corresponde al error cuadrático medio (MSE, *mean square error*). El MSE se calcula como el promedio de los cuadrados de las diferencias entre las predicciones del modelo y los valores reales. La expresión general de la función de costos es la siguiente:

$$L(x, g(f(x))) \, , \tag{2.16}$$

donde L es la función de costos, x son los datos de entrada, $f(x)$ corresponde a la información codificada, y $g(f(x))$ a la información reconstruida o decodificada.

Cuando la dimensionalidad de la capa oculta d es menor que la dimensionalidad de la entrada n, se dice que esta capa oculta se encuentra sub-completa. En caso contrario la capa oculta h esta sobre-completa, es decir, $d > n$.

En la Figura 2.5 se muestra un esquema de un Autoencoder básico. A este modelo entra un vector x por la capa de entrada, y se genera una representación codificada h, la cual pasa a ser decodificada por una capa de salida, obteniendo una versión reconstruida $\hat{x}$ del vector de entrada original.

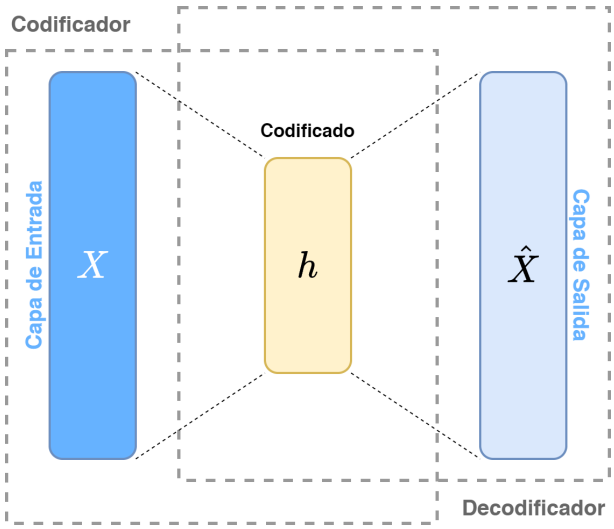


Figura 2.5: Esquema de componentes de un Autoencoder básico.

2.4. K-Means

El algoritmo K-Means es un método de agrupamiento no supervisado que busca encontrar grupos o clústeres en un conjunto de datos, donde cada clúster está representado por su centroide. El algoritmo sigue los siguientes pasos [29]:

1. Seleccionar el número de clústeres K que se desea obtener en el conjunto de datos.
2. Inicializar aleatoriamente las coordenadas de los centroides de cada clúster.
3. Asignar cada punto de datos al clúster cuyo centroide esté más cercano, utilizando una medida de distancia (generalmente la distancia euclidiana).
4. Recalcular los centroides de cada clúster tomando el promedio de las coordenadas de los puntos asignados a ese clúster.
5. Repetir los pasos 3 y 4 hasta que se cumpla un criterio de detención, como la convergencia de los centroides, la estabilidad de los clústeres o un número máximo de iteraciones.

El criterio de detención puede variar según la implementación o el problema específico. Dentro de los criterios de detención más utilizados se tienen los siguientes [30]:

- Los centroides dejan de cambiar luego de múltiples iteraciones del algoritmo, por lo que se asume convergencia.
- Los puntos dejan de cambiar de clúster, en vez de cambiar los centroides, los puntos se asignan a las mismas agrupaciones en cada iteración, por lo que se asume que terminó el entrenamiento.
- Se cumple el límite de iteraciones fijado para el entrenamiento.

Es importante destacar que K-Means es sensible a la inicialización aleatoria de los centroides y puede converger a óptimos locales en lugar de la solución global óptima. Para mitigar este problema, se pueden utilizar técnicas de inicialización mejoradas o ejecutar el algoritmo varias veces con diferentes inicializaciones y seleccionar la mejor solución.

2.5. K-Vecinos más Cercanos

El algoritmo de *k-vecinos más cercanos* (del inglés *k-nearest neighbors*), es un clasificador de aprendizaje supervisado no paramétrico [31], que utiliza proximidad para realizar clasificaciones o predicciones sobre un grupo de puntos individuales. Si bien puede ser utilizado tanto para regresiones como para problemas de clasificación, típicamente se utiliza como algoritmo de clasificación, asumiendo que puntos y observaciones similares se encuentran cercanas entre sí. Así, el objetivo de este algoritmo es identificar los vecinos cercanos para un determinado objeto, y poder asignar una etiqueta.

Para determinar cuales datos son más cercanos al objeto que se encuentra analizando, se utiliza la distancia entre el objeto consultado y el resto de puntos que se encuentran a su alrededor. Para esto, existen varias formas de medir distancia, dentro de las cuales se encuentran las siguientes:

- *Distancia Euclidiana*: corresponde a una medida de distancia o similitud entre dos puntos en un espacio euclidiano. Esta, se calcula mediante el teorema de Pitágoras para determinar la longitud del segmento en línea recta que conecta los dos puntos, como se muestra en la siguiente ecuación:

$$d(A, B) = \sqrt{\sum_{i=1}^{N} (b_i - a_i)^2} \,. \tag{2.17}$$

- *Distancia Coseno*: es una medida de similitud entre dos vectores en un espacio vectorial que se enfoca en la dirección y orientación de los vectores. Para esto, se calcula el coseno del ángulo entre los dos vectores, proporcionando valores entre 0 y 1, donde 0 indica que los vectores son ortogonales y no tienen similitud, y 1 indica que los vectores son idénticos o similares en dirección. La fórmula es la siguiente:

$$d(A, B) = 1 - \frac{A \cdot B}{\|A\| \cdot \|B\|} \,, \tag{2.18}$$

  donde A · B es el producto escalar de los vectores A y B, y $\|A\|$ y $\|B\|$ representan las normas (longitudes) de los vectores.

- *Distancia Mahalanobis*: es una medida distancia o de similitud entre dos puntos en un espacio multidimensional que considera tanto la dirección como la magnitud entre los puntos, y tiene en cuenta las correlaciones y las varianzas de las variables involucradas, cuya expresión es:

$$d(A, B) = \sqrt{(A - B)^T \cdot C^{-1} \cdot (A - B)} \,, \tag{2.19}$$

donde $(A-B)$ es la diferencia vectorial entre los dos puntos, C es la matriz de covarianza de las variables en el conjunto de datos y C^{-1} es la inversa de la matriz de covarianza.

El valor k en el algoritmo *KNN* define cuantos vecinos se deben verificar para determinar como clasificar el dato que se encuentra analizando. La elección de un valor adecuado de k depende del conjunto de datos y del problema específico [31]. Un valor pequeño de k (por ejemplo, k = 1) puede llevar a un sobreajuste, ya que el algoritmo se basará en una sola vecindad para realizar la clasificación. Por otro lado, un valor grande de k puede llevar a un subajuste, ya que se considerarán más vecinos y puede haber una dilución de la información.

La selección de un valor óptimo de k a menudo se realiza mediante técnicas de validación cruzada o mediante la búsqueda exhaustiva de diferentes valores de k y la evaluación de su rendimiento. Es importante encontrar un equilibrio entre la simplicidad del modelo (k pequeño) y la capacidad de generalización (k grande).

2.6. Detección de Outliers

2.6.1. Selección de Características en la Detección de Valores Atípicos

Es notoriamente difícil realizar la selección de características en la detección de valores atípicos debido a la naturaleza no supervisada del problema de detección de valores atípicos. A diferencia del problema de clasificación, donde se utilizan etiquetas como guía para el entrenamiento del modelo, es difícil aprender cómo las características se relacionan con la realidad (no observada) en la detección de anomalías no supervisados.

Una forma de medir la falta de uniformidad de un conjunto de puntos univariados (x_i, y_i) ; $i \in \{1, \ldots, N\}$ es la medida de curtosis (K, del inglés *Kurtosis*). El cálculo de la curtosis implica estandarizar los datos mediante la resta de la media μ y la división por la desviación estándar σ, para lograr obtener un conjunto de datos con una media de cero y una varianza unitaria, como sigue:

$$z_i = \frac{x_i - \mu}{\sigma} \, . \tag{2.20}$$

Se debe notar que en la ecuación 2.20, el promedio de los cuadrados de z_i siempre es igual a 1 por la definición de z_i. Una vez estandarizados los datos, la curtosis se calcula como el promedio de la cuarta potencia de z_i. De esta manera, para una distribución normal, el valor del estadístico es igual a 0, mientras que para que una curtosis positiva indica que los datos muestran valores atípicos más extremos que una distribución normal, y una curtosis negativa indica que la distribución de datos presenta valores atípicos menos extremos [32]. La curtosis se calcula de la siguiente manera:

$$K(z_i, \ldots, z_N) = \frac{\sum_{i=1}^{N} z_i^4}{N} \, . \tag{2.21}$$

Un posible uso para esta medida es combinar este cálculo con un método de enfoque *greedy* que iterativamente agregue características a un posible subconjunto S de características, con el objetivo de construir un subconjunto de características discriminativas con el valor más alto de curtosis multidimensional.

2.6.2. Análisis de Valores Extremos

El análisis de valores extremos (*EVA*) es una técnica que se utiliza para estudiar las colas de las distribuciones de datos y analizar eventos extremos o valores atípicos. En el análisis de valores extremos, se busca comprender y modelar la probabilidad y la magnitud de eventos raros que se encuentran en las colas de las distribuciones.

El enfoque principal del *EVA* es modelar y analizar las colas de las distribuciones, ya que estas zonas tienen una baja probabilidad de contener elementos. Por lo tanto, los elementos que se encuentran en estas colas se consideran como posibles anomalías o valores atípicos.

Si bien la distribución normal es la más fácil de analizar, ya que muchas pruebas estadísticas se basan en esta distribución, el análisis de valores extremos se puede aplicar a distribuciones arbitrarias. Las pruebas estadísticas, como el puntaje z, se utilizan para evaluar la significancia de los puntos de datos y determinar si se consideran atípicos o no. Aunque estas pruebas pueden no proporcionar una interpretación estadística precisa para distribuciones no normales, siguen siendo útiles como una heurística para identificar valores atípicos.

El problema de determinar las colas de las distribuciones y modelar los eventos extremos ha sido ampliamente estudiado en la literatura estadística [33]. Se han desarrollado diversas técnicas y métodos, como el método de bloques máximos y el enfoque de excedentes sobre el umbral, que permiten modelar y estimar los parámetros de las distribuciones de valores extremos.

En resumen, el análisis de valores extremos se centra en el estudio de eventos extremos y valores atípicos en las colas de las distribuciones de datos. Se utilizan diversas técnicas y métodos estadísticos para modelar y comprender la probabilidad y la magnitud de estos eventos raros.

2.6.3. Modelos Probabilísticos y Estadísticos

En los modelos probabilísticos y estadísticos, los datos son modelados en forma de una distribución de probabilidad, y se aprenden los parámetros del modelo. De esta manera, la clave en este tipo de modelos es la elección correcta de la distribución de los datos con la que se realiza el modelado [33].

Los parámetros de estas distribuciones se estiman a través de técnicas como la maximización de la esperanza (*EM*) en los datos observados para que la probabilidad o verosimilitud del proceso que genera los datos sea la mayor posible.

En cuanto a los valores atípicos, los modelos probabilísticos y estadísticos permiten de forma natural su identificación, debido a que los puntos de datos que tienen valores de ajuste muy bajos pueden ser considerados valores atípicos. Para ello, se pueden utilizar puntajes de ajuste, como los logaritmos de los valores de ajuste, que resaltan aquellos puntos de datos que se desvían significativamente del modelo.

Una de las principales ventajas de los modelos probabilísticos es que se pueden aplicar fácilmente a prácticamente cualquier tipo de datos, donde la elección correcta de la distribución de probabilidad es crucial, ya que diferentes distribuciones pueden capturar diferentes características de los datos. Por ejemplo, si los datos utilizados son categóricos, se puede utilizar distribuciones discretas como la distribución de Bernoulli o la distribución multinomial, para modelar cada componente.

Sin embargo, es importante tener en cuenta que este tipo de modelo, al asumir una distribución específica para los datos, puede limitar su aplicabilidad en ciertas situaciones. Además, a medida que aumenta el número de parámetros del modelo, se incrementa el riesgo de sobreajuste, obteniéndose un rendimiento deficiente para nuevos datos.

2.6.4. Modelos Lineales

Los modelos lineales, proyectan los datos a subespacios de menor dimensión mediante el uso de correlaciones lineales. El hiperplano (que corresponde a una recta para el caso bidimensional) que pasa a través de los puntos definidos por los datos, es determinado mediante el uso del análisis de regresión. Típicamente se utiliza mínimos cuadrados como medida de distancia al hiperplano de menor dimensión óptimo asociado al conjunto de datos utilizado.

El uso de la distancia de los puntos (datos) al hiperplano se utiliza para cuantificar el puntaje de anomalía de una observación, debido a que la distancia cuantifica la desviación de los datos normales, con respecto al modelo encontrado.

Para el caso bidimensional, a partir de los puntos del conjunto de datos (x_i, y_i) ; $i \in \{1, \ldots, N\}$, y el residuo ε_i que corresponde al error. Se crea el modelo:

$$y_i = a \cdot x_i + b + \varepsilon_i \ , \tag{2.22}$$

donde los valores de a y b son coeficientes que se aprenden de los datos al minimizar el error cuadrático medio, el cual se denota por $\sum_{i=1}^{N} \varepsilon_i^2$ [33].

2.6.5. Modelos Basados en Proximidad

Los modelos basados en proximidad asumen que un objeto es una anomalía si la proximidad del objeto a sus vecinos cercanos difiere significativamente de la proximidad de los otros datos a sus vecinos, para un mismo conjunto de datos [34]. Es por esto, que es importante considerar que el rendimiento de estos modelos depende de la elección adecuada de la medida de distancia o similitud utilizada, así como la configuración de parámetros.

Los métodos basados en proximidad se pueden aplicar en tres formas. La primera corresponde a métodos basados en el agrupamiento de datos (*Clustering*), métodos basados en la densidad de datos, y vecinos más cercanos [33].

2.6.6. Modelos teóricos de la información

Muchos de los modelos antes mencionados utilizan formas de resumen de datos para detectar valores atípicos. Estos resumenes pueden ser generados mediante parámetros de modelos probabilísticos generativos, como la media y la covarianza en el caso de la distribución normal multivariada, o mediante la creación de conglomerados o hiperplanos de representación de menor dimensión. Los modelos antes mencionados generan implícitamente un pequeño resumen de los datos, y las desviaciones de este resumen se marcan como valores atípicos.

Los modelos teóricos de la información realizan mediciones a partir del mismo principio, pero de una manera indirecta. La idea principal, es que los valores atípicos incrementan la longitud mínima necesaria para describir el conjunto de datos codificados, debido a los valores atípicos representan una desviación del intento de describir los datos. De esta manera, un conjunto de datos que presenta valores atípicos requiere de una mayor cantidad de parámetros, clusters, o patrones frecuentes con el objetivo de obtener un nivel de aproximación similar [33].

En consecuencia, en los métodos teóricos de la información se construye un libro de códigos que representa los datos, y los valores atípicos se definen como puntos cuya eliminación da como resultado la mayor disminución en la longitud de la descripción [35], o en una representación resumida más precisa en la misma longitud de descripción [36].

2.7. Modelo Base

2.7.1. Modelos basados en Características

One Class Support Vector Machine

La Máquina de Soporte Vectorial de una Clase (One-Class Support Vector Machine, SVM) es un modelo no supervisado para la detección de anomalías u *outliers*. A diferencia de la SVM supervisada tradicional, la SVM de una clase no utiliza etiquetas objetivo durante el proceso de entrenamiento del modelo. En cambio, aprende el límite para los puntos de datos normales e identifica los datos fuera de dicho límite como anomalías [37].

El concepto básico de la SVM de una clase es minimizar la hiperesfera que contiene los ejemplos de la clase única en los datos de entrenamiento, considerando que todos los demás ejemplos fuera de esta hiperesfera son *outliers* o se encuentran fuera de la distribución de los datos de entrenamiento.

La expresión matemática para calcular una hiperesfera con centro c y radio r es:

$$min\, r^2,\ \parallel \phi(x) - c \parallel^2 \leq r^2 \qquad \forall i = 1, 2, \ldots, n \tag{2.23}$$

La expresión anterior intenta minimizar el radio de una hiperesfera. Sin embargo, esta formulación es muy restrictiva para *outliers*. Por lo tanto, se utiliza una formulación más flexible para tolerar *outliers* en cierta medida:

$$min\, r^2 + \frac{1}{\nu n} \sum_{i=1}^{n} \xi_i,\ \parallel \phi(x) - c \parallel^2 \leq r^2 + \xi_i \qquad \forall i = 1, 2, \ldots, n \tag{2.24}$$

Donde la función ϕ es la transformación hiperesférica de las muestras x. La figura 2.6 muestra la formulación de una hiperesfera al minimizar el radio r y el centro c.

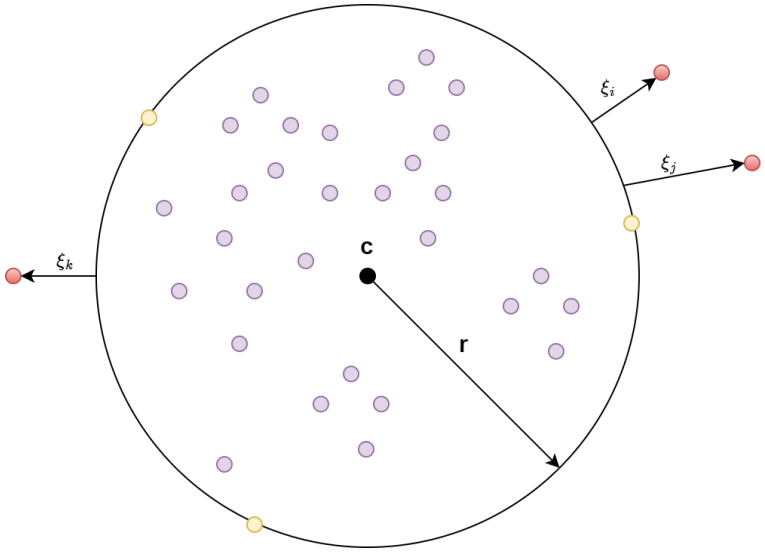


Figura 2.6: Hiperesfera de centro c y radio r, con *outliers* de color rojo fuera del radio.

Isolation Forest

Isolation Forest es un método de detección de anomalías no supervisado que no utiliza observaciones clasificadas para el entrenamiento. Su funcionamiento es similar al de *Random Forest*, pero con la diferencia de que las particiones para cada árbol de realizan de forma aleatoria [38].

Para decidir si una observación es anómala o no, para cada observación se calcula su puntaje de anomalía según la siguiente expresión:

$$s(x, n) = 2 - E(h(x)) \cdot c(n)\,, \tag{2.25}$$

donde $h(x)$ es la profundidad o altura media de x de los *Isolation Tree* construidos, $c(n)$ es la altura media para encontrar un nodo en un árbol, y n es el tamaño del conjunto de datos. De esta manera, si el valor s es cercano a 1, es probable que la observación corresponda a una anomalía, mientras que si el valor de s es inferior a 0.5, es probable que el objeto estudiado no corresponda a una anomalía.

Local Outlier Factor

El algoritmo *Local Outlier Factor* (LOF) es un método de detección de anomalías no supervisado que calcula la desviación de densidad local de un punto de datos en comparación con sus vecinos. Considera como outliers las muestras que tienen una densidad sustancialmente más baja que la de sus vecinos. LOF es capaz de detectar anomalías en conjuntos de datos sin necesidad de datos etiquetados, lo que lo convierte en una herramienta versátil y efectiva para identificar outliers en diversos dominios [39].

El algoritmo Local Outlier Factor (LOF) utiliza la desviación de densidad local (*Local Reachability Density, LRD*) para compararla con la densidad promedio de sus K vecinos. De esta manera, LOF es la relación entre la densidad promedio de los K vecinos de un punto A y la densidad de A [40]. Esta se calcula como:

$$LOF_k(A) = \frac{\sum_{Xj \in N_k(A)} LRD_k(Xj)}{\parallel N_k(A) \parallel} \times \frac{1}{LRD_k(A)} , \tag{2.26}$$

donde la desviación de densidad local (LRD) es el inverso de la distancia de alcance promedio desde A hacia sus vecinos. Es una medida intuitiva que indica qué tan lejos se encuentra un punto de su grupo de vecinos más cercano. Valores bajos de LRD implican que el punto se encuentra lejos del grupo de puntos más cercano. La LDR se computa como sigue:

$$LRD_k(A) = \frac{1}{\sum_{\sum_{Xj \in N_k(A)} \frac{RD(A,Xj)}{\|N_k(A)\|}}} . \tag{2.27}$$

El alcance (*Reachability Distance, RD*) se define como el máximo entre la distancia K entre Xj y la distancia entre Xi y Xj. La medida de distancia utilizada puede ser específica del problema (Euclidiana, Manhattan, etc.). El alcance se define de la siguiente manera:

$$RD(Xi, Xj) = max(K - distance(Xj), distance(Xi, Xj)) . \tag{2.28}$$

2.7.2. Modelos basados en Redes Neuronales

Time-Frequency Consistency

El modelo *Time-Frequency Consistency* (TF-C) [2] utiliza aprendizaje contrastivo auto-supervisado para transferir conocimientos entre dominios de series temporales. Este enfoque se basa en la dualidad fundamental entre las vistas de tiempo y frecuencia de las señales temporales. Para esto incorpora representaciones aprendidas tanto en el dominio del tiempo como en el dominio de la frecuencia de una misma muestra de serie temporal, de manera que en el espacio conjunto de tiempo-frecuencia estas representaciones estén cercanas entre sí, pero más distantes si se asocian con muestras de diferentes series temporales.

TF-C consta de cuatro componentes de red neuronal:

- Codificador contrastivo de tiempo.
- Codificador contrastivo de frecuencia.
- Dos proyectores de espacios cruzados que mapean las representaciones de tiempo y frecuencia al mismo espacio tiempo-frecuencia.

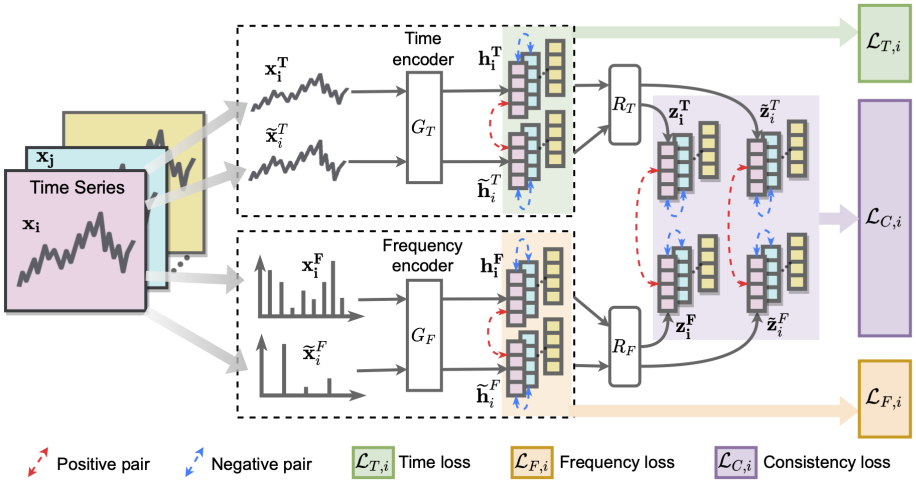


Figura 2.7: Esquema de componentes de red neuronal, para el modelo TF-C. Tomado de [2].

Para posicionar las representaciones de tiempo y frecuencia cerca una de la otra, TF-C especifica el siguiente objetivo de aprendizaje contrastivo:

1. Aplicar aumentaciones en el dominio del tiempo a la serie temporal de entrada. Las representaciones originales y las vistas aumentadas, producidas por el codificador de dominio del tiempo, se utilizan para calcular un término en la función de pérdida contrastiva.
2. Transformar la serie temporal de entrada a su espectro de frecuencia. Luego, se aplican aumentaciones en el dominio de la frecuencia a la muestra de serie temporal, se producen las representaciones y se calcula otro término en la pérdida contrastiva.

3. TF-C requiere la consistencia entre las representaciones en el dominio del tiempo y en el dominio de la frecuencia, lo que se logra mediante un elemento de pérdida de consistencia en el espacio de tiempo-frecuencia. De esta manera, la pérdida contrastiva total corresponde a la suma ponderada de estos tres términos.

4. Durante la fase de ajuste fino (*fine-tuning*), se concatenan las representaciones de dominio del tiempo y de dominio de la frecuencia para formar la representación de la muestra.

Time-Series representation learning framework via Temporal and Contextual Contrasting

TS-TCC (*Time-Series representation learning framework via Temporal and Contextual Contrasting*) [3] es un modelo de aprendizaje auto-supervisado para series de tiempo que utiliza contrastes temporales y contextuales para aprender representaciones efectivas. Genera dos vistas de cada muestra de serie de tiempo mediante aumentos débiles y fuertes. A través del módulo de contraste temporal, aprende relaciones sólidas entre las vistas, capturando patrones temporales. Además, el módulo de contraste contextual permite aprender representaciones discriminativas mediante la predicción cruzada de pasos de tiempo futuros. De esta manera, TS-TCC logra representaciones robustas y altamente informativas para datos no etiquetados de series de tiempo.

A continuación se enumera el funcionamiento del modelo TS-TCC, el cual se encuentra resumido en la Figura 2.8:

1. Se toma una muestra x y se generan dos vistas aumentadas: una mediante un aumento fuerte x^s (permutation-and-jitter) y otra con un aumento débil x^w (jitter-and-scale). Estos aumentos ayudan a enriquecer la información y proporcionar diferentes perspectivas de la misma muestra.

2. Las vistas aumentadas x^s y x^w se pasan a través de un encoder convolucional que consta de 3 bloques. Cada bloque realiza operaciones de convolución y activaciones no lineales para extraer representaciones latentes z^s y z^w de alta dimensión para cada vista.

3. Las representaciones latentes z^s y z^w se introducen en un módulo de contraste temporal. Este módulo utiliza un modelo autorregresivo para desplegar una pérdida de contraste que tiene como objetivo extraer características temporales relevantes en el espacio latente.

4. Se realiza una predicción cruzada usando el contexto c_i obtenido a partir del aumento fuerte c^s. El contexto c_i se utiliza para predecir los pasos de tiempo futuros en las representaciones del aumento débil z^w_{t+k} y viceversa. Esto permite que el modelo aprenda a capturar relaciones temporales entre los diferentes puntos de tiempo en las series de tiempo.

5. Para predecir los pasos de tiempo futuros, se utiliza un modelo logarítmico bilineal. Este modelo preserva la información mutua entre las representaciones z_{t+k} y c_t, asegurando que la información temporal se mantenga en el espacio latente.

6. Para cada lote de N muestras de entrada, se generan dos contextos para cada muestra a partir de las dos vistas aumentadas, lo que da lugar a un total de $2N$ contextos. Estos contextos representan diferentes perspectivas y representaciones de las mismas muestras.

7. Se calcula la pérdida de contraste para cada contexto. La pérdida de contraste es una medida de similitud entre el contexto y su muestra positiva c_t^w en comparación con todas las demás muestras $(2N-1)$, incluido el par positivo y los pares negativos $(2N-2)$. Esta normalización de la pérdida asegura una comparación adecuada entre los contextos y su relevancia en el espacio latente.

8. La pérdida total del algoritmo TF-C es una combinación ponderada de las dos pérdidas de contraste temporal y la pérdida de contraste contextual. Esta pérdida total permite extraer tanto características temporales como contextuales de las series de tiempo en el espacio latente, lo que mejora el rendimiento del modelo en tareas posteriores, como la clasificación o detección de anomalías.

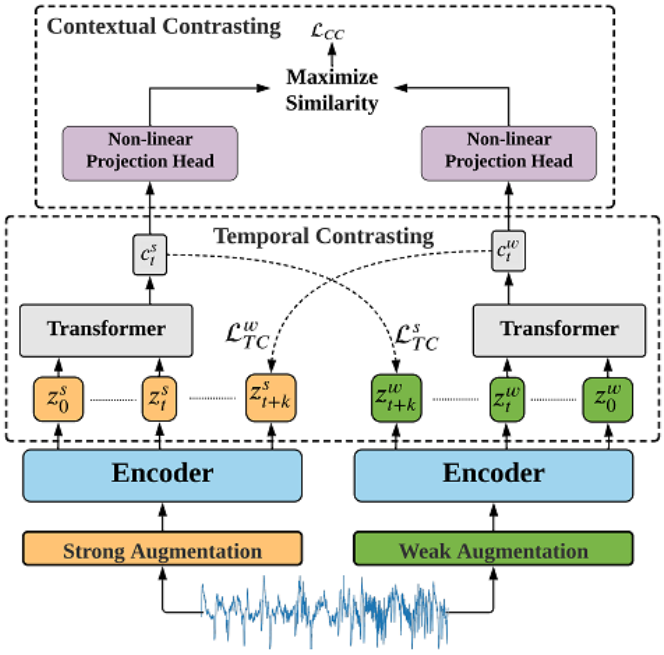


Figura 2.8: Esquema de componentes de red neuronal, para el modelo TS-TCC. Tomado de [3].

Multi-Class Deep SVDD

El modelo MCDSVDD modela múltiples hiperesferas, a diferencia de otros modelos donde solo se modelo una [41]. Cada una corresponde a una clase específica. Se espera que los objetos de la misma clase estén cercanos entre sí y alejados de objetos de clases diferentes. Dado que las muestras anómalas provienen de clases no vistas, sus distancias a cada hiperesfera deberían ser mayores que las de los puntos de datos normales. Este enfoque se denomina Multi-Class Deep SVDD (MCDSVDD).

Siguiendo este enfoque, es posible definir una puntuación de anomalía basada en la distancia de los puntos de datos a los centros de las hiperesferas.

Similar a Deep SVDD, se entrena un Autoencoder (AE) hasta la convergencia y se elimina el decodificador. Suponiendo pares de datos normales provenientes de M clases diferentes $y \in \{1, \ldots, M\}$, el centro de cada hiperesfera se estima como:

$$c_j = \frac{1}{N_j} \sum_{i=1}^{N} \mathbb{1}_{(y_i = y_j)}(x_i; \theta^*_{AE}) \,, \tag{2.29}$$

donde θ^*_{AE} son los parámetros del AE entrenado, $\mathbb{1}_{(y_i=y_j)}$ es una función indicatriz que es 1 si $y_i = j$ y 0 de lo contrario, y N_j es el número de puntos de datos que pertenecen a la clase j.

Utilizando θ^*_{AE} como parámetros preentrenados, se reoptimiza el vector de parámetros θ del codificador de la siguiente manera:

$$\min_{\theta} \sum_{j=1}^{M} \frac{1}{N_j} \sum_{i=1}^{N} \mathbb{1}_{(y_i=y_j)} \|E(x_i; \theta^*_{AE}) - c_j\|^2 + \frac{\lambda}{2} \theta^\tau \theta \,, \tag{2.30}$$

donde λ es un hiperparámetro que controla el regularizador de decaimiento de peso en los parámetros θ de la red.

Durante la prueba, la puntuación anómala $A(\cdot)$ se determina midiendo la distancia de cada punto de datos al centro de su hiperesfera más cercana, como sigue:

$$A(x_i) = \min_{j} \|E(x_i; \theta^*) - c_j\|^2 \,, \tag{2.31}$$

donde θ^* corresponde a los parámetros de la red neuronal entrenada.

La implementación de MCDSVDD realiza validación cruzada de la función de pérdida no supervisada sobre un conjunto de validación compuesto por *inliers*.

Capítulo 3

Metodología

3.1. Modelo Propuesto

El modelo propuesto para la detección de anomalías astronómicas corresponde a un modelo basado en aprendizaje contrastivo mediante aprendizaje auto-supervisado, el cual debe ser capaz de procesar series de tiempo multidimensionales, muestreadas de forma irregular y de largo variable dentro de un mismo conjunto de datos. Para esto, se propone utilizar, en primera instancia, una red neuronal (Autoencoder) que codifique las curvas de luz en vectores de largo fijo, mediante aprendizaje no supervisado, para luego utilizar los vectores de características obtenidos a partir de este modelo, para entrenar un modelo MLP, mediante técnicas de aprendizaje auto-supervisado, para detectar curvas de luz anómalas.

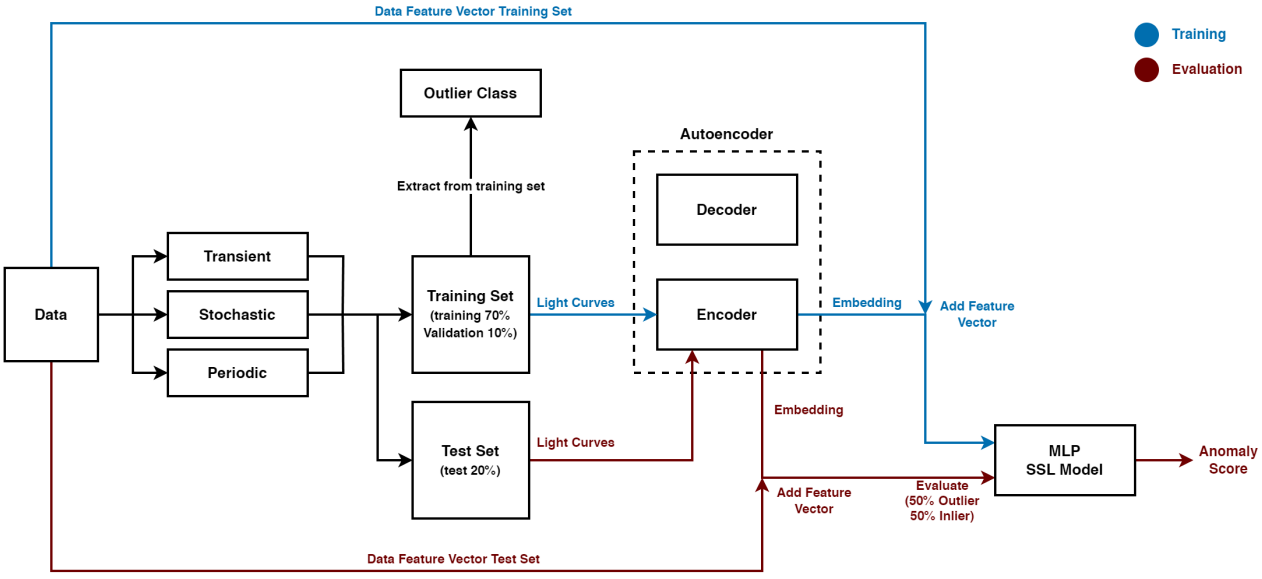


Figura 3.1: Metodología para el entrenamiento y evaluación del algoritmo de detección de anomalías. El flujo representado en el diagrama se repite por cada una de las clases del conjunto de datos, siendo consideradas como la clase *outlier* al momento de ser extraídas del conjunto de entrenamiento.

La metodología antes mencionada se describe en la Figura 3.1, donde los datos se separan en conjunto de entrenamiento y prueba. Una vez separados los conjuntos, las curvas de luz asociadas a la clase considerada como outlier se extraen del conjunto de entrenamiento, para posteriormente codificar todas las curvas de luz restantes a un largo fijo, mediante el uso de un Autoencoder. Luego, a la codificación obtenida por el autoencoder, se le añade un vector de características asociado a cada objeto astronómico, para entrenar el modelo MLP, mediante aprendizaje auto-supervisado, para la asignación de puntajes de anomalía. Los detalles específicos de los modelos utilizados y los métodos de asignación de puntajes se describen en las secciones siguientes.

3.1.1. Codificador

El modelo utilizado para la codificación y extracción de características de las curvas de luz es un Autoencoder compuesto por dos capas convolucionales seguidas de dos capas recurrentes LSTM bidireccionales. La arquitectura del modelo se detalla en el Anexo C.1.

Durante el entrenamiento, el modelo pasa por una etapa de codificación y decodificación de las curvas de luz. En la etapa de codificación, las curvas de luz se pasan por las capas convolucionales y recurrentes LSTM bidireccionales para extraer características esenciales. Luego, en la etapa de decodificación, el modelo trata de reconstruir las curvas de luz originales a partir de las características extraídas.

La función de costos utilizada para medir la calidad de la reconstrucción es el error cuadrático medio *MSE* (*Mean Square Error Loss*). Esta función calcula la diferencia cuadrada entre cada elemento de la secuencia reconstruida y su correspondiente elemento en la secuencia original.

El *MSE Loss* se calcula mediante la siguiente fórmula:

$$MSELoss = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2\,, \tag{3.1}$$

donde N es el número total de elementos en la secuencia, y_i es el valor del elemento en la secuencia original y $\hat{y}_i$ es el valor del elemento en la secuencia reconstruida por el modelo autoencoder.

Al minimizar el *MSE Loss* durante el entrenamiento, el modelo busca ajustar sus parámetros para que la secuencia reconstruida se acerque lo más posible a la secuencia original, lo que indica que el modelo está aprendiendo a representar de manera efectiva la información presente en las curvas de luz, y por consiguiente, el modelo mejora su capacidad para extraer características relevantes.

En la sección 4.1, se llevan a cabo comparaciones entre tres enfoques: el primero utiliza un autoencoder (AE) antes del entrenamiento del modelo de aprendizaje auto-supervisado (clasificador), el segundo solo utiliza el vector de características, y el tercero en el que la codificación y asignación de puntajes de anomalía se realizan directamente por el modelo de aprendizaje auto-supervisado (clasificador), sin la codificación previa de las curvas de luz a través del modelo AE.

3.1.2. Clasificador

El clasificador utilizado corresponde a un modelo Perceptron Multicapa (*MLP*, por sus siglas en inglés, *Multilayer Perceptron*). Este recibe como datos de entrada las curvas de luz codificadas de longitud fija mediante el Autoencoder mencionado anteriormente, junto con un nuevo canal que contiene un vector de características asociadas a la banda utilizada en este trabajo (banda g) para cada objeto astronómico. Estas características corresponden a un subconjunto de los 169 atributos utilizados en [42], que varía según el conjunto de datos utilizado. Esto último se detalla en la sección 3.5 donde se describen los conjuntos de datos utilizados. La arquitectura completa del modelo se muestra en el Anexo C.2.

El clasificador *MLP* toma en cuenta tanto la información de las series de tiempo codificadas como las características adicionales. Esto permite aprovechar tanto las características generadas por el Autoencoder como las características específicas seleccionadas para cada objeto astronómico.

Se utiliza la función de pérdidas *infoNCE Loss* (pérdida de entropía cruzada normalizada por información, por sus siglas en inglés) [6]. Esta función es comúnmente utilizada en modelos de aprendizaje auto-supervisado para entrenamiento de representaciones aprendidas. Esta proporciona una medida de similitud entre pares de ejemplos de entrada y mide la capacidad del modelo para discriminar entre pares positivos y negativos mediante una matriz de similitud, sin hacer uso de las etiquetas de los datos, y tampoco se seleccionan de forma manual los pares positivos y negativos a la hora de entrenar el modelo mediante aprendizaje constrastivo [4].

Al igual que en el entrenamiento realizado para SimCLR [9], para un *minibatch* de tamaño N, se define la tarea de predicción en pares de versiones aumentadas de una cierta muestra $(2N)$, es decir, no se seleccionan los pares negativos, considerando las $2(N-1)$ pares restantes como muestras negativas. De esta manera la función de costos se define como en la siguiente ecuación:

$$l_{i,j} = -\log \frac{exp(sim(z_i, z_j)/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} exp(sim(z_i, z_j)/\tau)} , \tag{3.2}$$

donde $sim(u, v) = \frac{u^T v}{\|u\|\|v\|}$ denota al producto punto entre los vectores u y v con normalización l_2, es decir similitud de coseno. $\mathbb{1}_{[k \neq i]}$ corresponde a una función indicatriz, de tal manera que tiene valor igual a 1 solo si $k = i$ y τ denota un parámetro de temperatura. Por último, la pérdida total corresponde a todos los pares positivos, ambos (i,j) y (j,i), en un *minibatch*.

El modelo MLP cuenta con diferentes configuraciones de hiperparámetros, los cuales están resumidos en la Tabla 3.1. Estos hiperparámetros incluyen el número de capas ocultas, el tamaño de las capas ocultas, la tasa de aprendizaje, el número de épocas de entrenamiento y otros parámetros específicos del modelo. La búsqueda de hiperparámetros y la definición de cada uno de estos se detalla en la sección 3.6.1.

Tabla 3.1: Hiperparámetros y sus respectivos valores utilizados para el entrenamiento del modelo MLP de aprendizaje auto-supervisado.

Hiperparámetros	Rango de Valores
Input Size	[34, 39, 78, 169]
Hidden Size	[16, 32, 64, 128]
Output Size	[4, 10, 15, 20, 25]
Batch Size	[256, 1024, 2046, 3000]
Learning Rate	[0.001, 0.0001, 0.0001]
Loss Function	infoNCE
Similarity Distance	[euclidian, cosine, mahalanobis, knn(euc)]

3.1.3. Asignación de Puntaje

Para determinar si un objeto corresponde a una anomalía, se utiliza un modelo de red neuronal (en este caso el clasificador MLP de la sección 3.1.2) que genera un vector de características para cada objeto. El rango de dimensiones de este vector corresponde al rango que se muestra en el *Output Size* de la Tabla 3.1. El modelo asigna un puntaje de anomalía al objeto basado en su distancia al clúster C_k más cercano en el espacio generado por la red neuronal. El número de clústeres se determina según el número de clases presentes en el conjunto de datos no anómalos (*inlier*).

Para encontrar representaciones para cada clúster asociado a una clase *inlier* se utilizan dos métodos diferentes. El primer método utiliza el algoritmo *K-Means*, descrito en la sección 2.4, para encontrar los k centroides respectivos de cada clase presente en el conjunto *inlier*. El segundo método forma representaciones mediante el promedio de los vectores de características generados por la red neuronal MLP, además del cálculo de covarianza.

Se utilizan diferentes medidas de distancia para asignar puntajes de anomalía. Con el primer método, se utiliza la distancia euclidiana o el coseno entre la representación generada por la red neuronal y el centroide más cercano. Esta distancia se utiliza como puntaje de anomalía, donde objetos muy lejanos a las representaciones de las clases *inlier* se consideran *outliers*. Con el segundo método, se utiliza la distancia Mahalanobis junto con las matrices de covarianza formadas para cada clase *inlier*, calculando la distancia del objeto sujeto a análisis a los centroides más cercanos.

Adicionalmente, se emplea el algoritmo *KNN* (ver sección 2.5) para etiquetar los datos basándose en los vecinos más cercanos, utilizando la distancia euclidiana como medida de similitud. A partir de las representaciones obtenidas por el algoritmo, se asigna un puntaje de anomalía basado en la distancia y la relación con las clases *inlier*.

3.1.4. Test Estadístico

Se llevó a cabo un test estadístico de permutación (permutation-test) [43] con el propósito de comparar el rendimiento del modelo de aprendizaje auto-supervisado propuesto (SSL) con los modelos de referencia, incluyendo tanto modelos basados en características como redes neuronales. En este contexto, se establecieron dos hipótesis:

- Hipótesis Nula (H_0): No existe una diferencia significativa en el rendimiento promedio entre el modelo SSL y los modelos de referencia.
- Hipótesis Alternativa (H_1): Existe una diferencia significativa en el rendimiento promedio entre el modelo SSL y los modelos de referencia.

Para llevar a cabo esta evaluación, se calculó la diferencia absoluta entre los puntajes asignados por los modelos en cuestión, utilizando un número fijo de permutaciones (en este caso, 10,000 permutaciones). Posteriormente, se analizó si la diferencia observada en los puntajes era estadísticamente significativa en comparación con las permutaciones generadas aleatoriamente.

Este enfoque se mostró útil para evaluar si el modelo SSL superaba de manera significativa a los modelos de referencia, y si dicha diferencia estaba respaldada de manera estadística por los datos. Si la diferencia observada en los puntajes resultaba sustancialmente mayor de lo que podría esperarse por azar, entonces se podía rechazar la hipótesis nula (H_0) y concluir que el modelo SSL exhibía un rendimiento significativamente superior.

3.2. Métricas de Evaluación

Para evaluar los modelos base y el propuesto, se propone utilizar como criterio de evaluación tanto el área bajo la curva de precisión y exhaustividad (*AUCPR*, por sus siglas en inglés, *Area Under Precision Recall Curve*) como el área bajo la curva *ROC* (*AUROC*, por sus siglas en inglés, *Area Under Receiver Operating Characteristic Curve*) [2][10]. Estas métricas de rendimiento son especialmente útiles cuando se trabaja con conjuntos de datos desequilibrados y se centra en la detección de ejemplos positivos o anomalías.

La curva de precisión y exhaustividad muestra la relación entre la precisión y la exhaustividad (*recall*) para diferentes valores de umbral, mientras que la curva ROC muestra la relación entre la tasa de verdaderos positivos (*TPR*, *True Positive Rate*) y la tasa de falsos positivos (*FPR*, *False Positive Rate*) para diferentes umbrales. La precisión, la exhaustividad, la *TPR* y la *FPR* son medidas que ayudan a evaluar el rendimiento de los modelos en diferentes aspectos.

El *AUCPR* y el *AUROC* miden el área bajo las respectivas curvas y proporcionan una medida agregada del rendimiento del modelo para un rango de umbrales. El *AUCPR* se enfoca en la precisión y la exhaustividad, mientras que el *AUROC* se centra en la *TPR* y la *FPR*. El uso de ambas métricas permite evaluar el rendimiento del modelo desde diferentes

perspectivas y considerar diferentes puntos de equilibrio entre precisión y exhaustividad, así como entre la tasa de verdaderos positivos y la tasa de falsos positivos.

Al utilizar el *AUCPR* y el *AUROC*, se evita la necesidad de fijar un umbral específico y se evalúa el rendimiento del modelo de manera más completa. Esto es especialmente beneficioso cuando se trabaja con conjuntos de datos desequilibrados y se busca encontrar ejemplos positivos o anomalías.

3.3. Preprocesamiento de los Datos

3.3.1. Curvas de Luz

Los datos utilizados son curvas de luz en la banda de observación g, siguiendo la metodología utilizada en [4]. Adicionalmente se realizaron pruebas utilizando tanto la banda de observación g como la de observación r, cuyos resultados se presentan en el Capítulo 4. Estas curvas de luz tienen tres canales: el primer canal corresponde al tiempo, medido en fecha juliana modificada (*Modified Julian Dates*) que representa el número de días y fracciones desde la medianoche del Tiempo Universal del 17 de noviembre de 1858. El segundo canal corresponde a la magnitud aparente del objeto, y el tercer canal contiene el error de la magnitud.

El preprocesamiento de las curvas de luz se realiza para que tengan un promedio de cero y una varianza unitaria. Esto se logra de la siguiente manera:

$$m_{norm} = \frac{m_i - \mu}{\sigma} \,, \tag{3.3}$$

$$\sigma^m_{norm} = \frac{\sigma^m_i}{\sigma} \,, \tag{3.4}$$

donde m_{norm} es la serie de magnitud normalizada, m_i la serie de magnitud original, μ el promedio temporal de la magnitud, y σ corresponde a la desviación estándar de la magnitud. En cuanto al error de magnitud, en la ecuación 3.4 el valor σ^m_{norm} representa el error de magnitud normalizado, y σ^m_i es la serie de tiempo del error de magnitud original.

Para el canal de tiempo, se realiza una normalización específica. Primero, se resta el valor del tiempo inicial a todos los puntos de tiempo en la serie. Luego, se aplica una escala logarítmica a la serie de tiempo para normalizarla. Esto se hace debido a que algunos tiempos de muestreo tienen órdenes de magnitud altos. El valor del tiempo inicial se define como cero, teniendo en cuenta que la escala logarítmica retornará $-\infty$ para el tiempo de la primera muestra.

De esta manera, el preprocesamiento de las curvas de luz garantiza que estén en una escala común y facilita su posterior análisis y modelado.

3.3.2. Vector de Características

En conjunto a las curvas de luz, se emplean 78 características que se derivan de las curvas de luz de la banda de observación *g*. Estas características suelen contener una cantidad considerable de valores nulos. Para abordar esta situación, se optaron por dos enfoques:

- Eliminación y Relleno de Valores Nulos: En primer lugar, se descartaron las características que contenían una cantidad de valores nulos por encima de un umbral específico. Luego, se procedió a reemplazar los valores nulos restantes con una constante, en este caso, se utilizó el valor cero.
- Transformación Basada en Cuantiles: La segunda opción implicó una transformación que utiliza información de los cuantiles de los datos. Esta transformación normaliza los datos para que estén en un rango entre 0 y 1. De esta manera, la distribución de los datos se asemeja a una distribución normal, lo que reduce el impacto de los valores atípicos (*outliers*) en los datos.

  Después de aplicar estas transformaciones, se agregó un valor pequeño (0.1) a cada característica, con el fin de evitar que la normalización afectara significativamente los datos. Además, se reemplazaron los valores nulos por el valor 0.0 en el conjunto de datos.

Para las pruebas realizadas al utilizar ambas bandas de observación *g* y *r*, se utilizó un total de 169 características, de las cuales 78 derivan de la banda de observación *g*, 78 de la *r*, y las 13 restantes son computadas utilizando ambas bandas en conjunto. LA descripción de estas 169 características se encuentra en el Anexo B.

3.4. Transformaciones

Si bien, durante el entrenamiento del modelo solo se utilizan las clases que no son consideradas como anomalía, se utilizan transformaciones a las series de tiempo, con el objetivo de introducir ruido o variabilidad, permitiendo al modelo tener una mayor incertidumbre, técnica conocida como aumento de datos (en inglés *Data Augmentation*).

El aumento de datos es una estrategia clave en este proceso. Esta técnica implica la generación de nuevas instancias de series de tiempo a partir de las existentes mediante la aplicación de transformaciones específicas. Estas transformaciones se describen a continuación:

3.4.1. Inversión de Amplitud

La Inversión de Amplitud consiste en invertir los valores del canal de magnitud de la serie de tiempo como se muestra en la siguiente expresión:

$$x' = -x_i \,, \tag{3.5}$$

$$i = 1, \ldots, n \,. \tag{3.6}$$

3.4.2. Inversión de Tiempo

Consiste en invertir temporalmente los elementos de la curva de luz. Esto solamente aplica para los canales de magnitud y error de magnitud de la curva, como se muestra en la siguiente expresión:

$$x' = x_n, x_{n-1}, \ldots, x_1 \,. \tag{3.7}$$

3.4.3. Desplazamiento Temporal

El desplazamiento temporal (*time shifting*) de una serie de tiempo consiste en desplazar la curva hacia el futuro, realizando una rotación circular de los elementos del vector, ingresando los valores de la serie que salen por el desplazamiento, al comienzo de la secuencia. El desplazamiento de la curva se realiza de forma aleatoria.

3.4.4. Escalamiento Aleatorio

El escalado aleatorio (*random scaling*) consiste en cambiar la magnitud de ciertas partes de la serie de tiempo, pero manteniendo la forma general de la señal a medida que se cambian los valores de esta, como se muestra en la expresión a continuación:

$$x'(\alpha) = \left\{\alpha \cdot x_1, \alpha \cdot x_2, \ldots, \alpha \cdot x_n\right\} \,, \tag{3.8}$$

donde el valor $\alpha > 0$ corresponde al factor de escalado que se aplica a los datos de la muestra. El valor de α se elige de forma aleatoria en un rango entre $[0.8, 1.2]$ cada vez que se aplica esta transformación durante el entrenamiento.

A pesar de que los datos se normalizan previamente en un rango de 0 a 1, se ha observado que en ciertas clases, escalar los datos ligeramente por encima de la normalización produce resultados satisfactorios, como se detalla en capítulos posteriores.

3.4.5. Deformación de Magnitud

El deformado de magnitud (*magnitude warping*) es una técnica utilizada que consiste en la aplicación de un escalado variable a diferentes puntos de la curva de datos. Para definir dónde aplicar la transformación se define un conjunto de nodos $u = u_1, \ldots, u_n$ que representan el paso en el que se realiza el escalado y sus valores se generan utilizando una distribución normal. Luego, se define la magnitud del escalado mediante una interpolación cúbica spline de estos nodos $S(x)$ [44]. La interpolación cúbica spline es un enfoque matemático utilizado para crear una curva suave que pasa a través de un conjunto de puntos de datos.

Por ultimo, la deformación de la magnitud se puede definir como en la siguiente expresión:

$$x'(\alpha) = \left\{\alpha_1 \cdot x_1, \alpha_2 \cdot x_2, \ldots, \alpha_n \cdot x_n\right\}, \tag{3.9}$$

donde el vector α se define $\alpha = \alpha_1, \ldots, \alpha_n = S(x)$.

3.4.6. Deformación del Tiempo

El deformado temporal (*time warping*) es similar al caso de magnitud, con la diferencia de que el deformado temporal modifica la curva con respecto a la dimensión temporal. Es decir, en vez de variar la magnitud de la serie de tiempo en cada paso, estrecha y acorta el tiempo en segmentos de la señal. Al igual que para el deformado de magnitud, se utiliza interpolación cúbica spline S(u) entre diferentes nodos generados a partir de una distribución normal como se muestra en la siguiente expresión:

$$x'(\tau) = \left\{x_\tau(1), \ldots, x_\tau(t), \ldots, x_\tau(T)\right\}, \tag{3.10}$$

donde τ determina la magnitud del deformado temporal.

3.4.7. Corte de Tiempo

El corte de tiempo o *time slicing*, consiste en cortar una pequeña porción de cada muestra de datos, para generar una vista aumentada que sea diferente a la original, conteniendo solo una parte del total de la serie [44].

Una de las inconveniencias de este tipo de transformación en series de tiempo, es que puede producir muestras sintéticas que sean inválidas, debido al corte de características importantes de la muestra de datos.

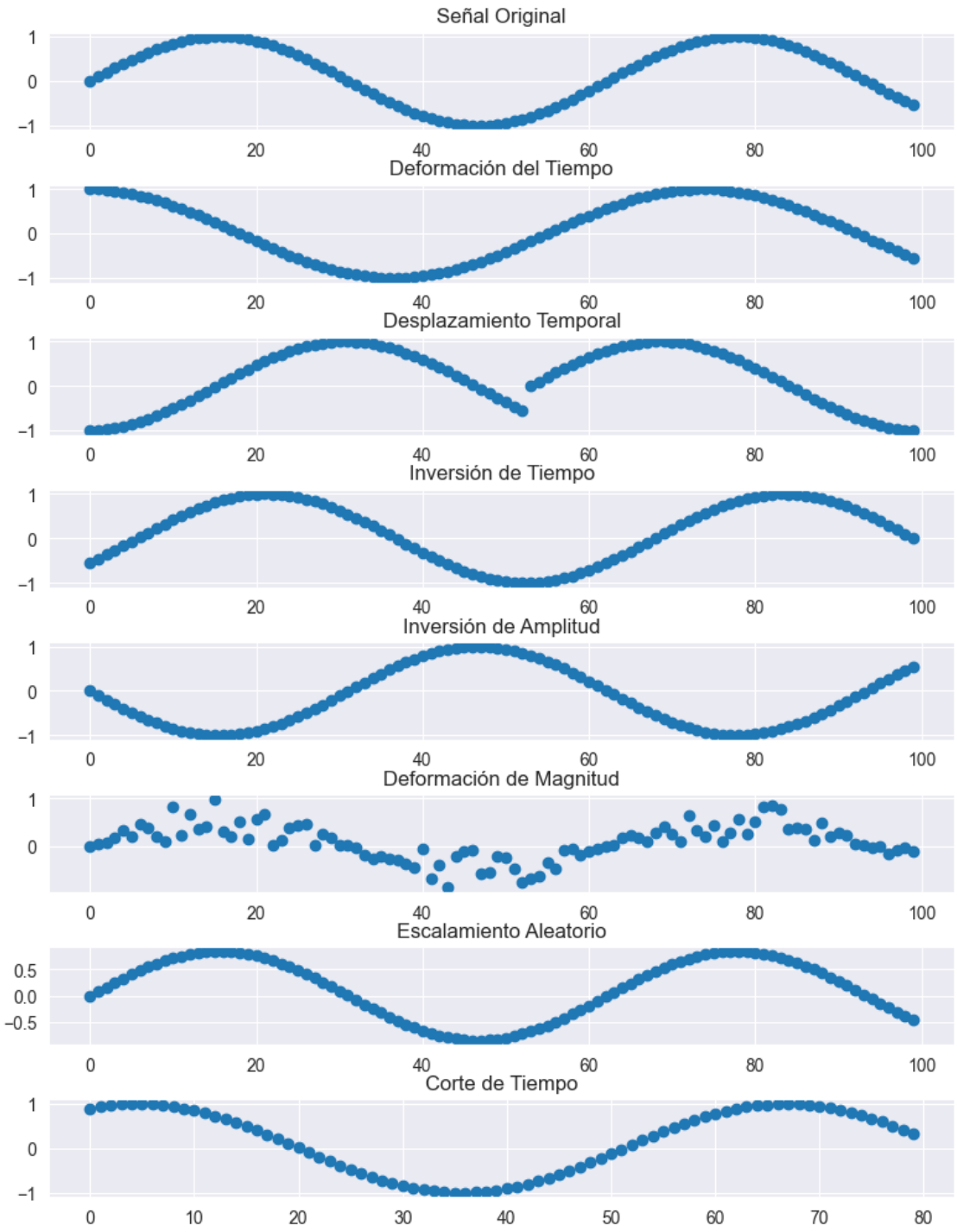


Figura 3.2: Ejemplo de la aplicación de distintas transformaciones sobre una señal sinusoidal.

3.5. Conjunto de Datos

El conjunto de datos utilizado corresponde al conjunto *Zwicky Transient Facility* (ZTF), el cual cuenta con curvas de luz de objetos astronómicos recopilados y curados por el *broker ALeRCE* [42][45][46] .

El conjunto ZTF utilizado se encuentra compuesto por 15 clases que se pueden dividir en objetos astronómicos transientes (ZTF-TRA), estocásticos (ZTF-EST) y objetos periódicos (ZTF-PER). En este trabajo se utiliza principalmente la banda de observación g, al igual que en [4], junto a un subconjunto de las 78 características asociadas a la banda de observación g (ver descripción en el Anexo B), de las 169 descritas y utilizadas en [42] las cuales contemplan tanto la banda de observación g como la banda r. La cantidad de características varía dependiendo del conjunto de datos con el que se esta trabajando, debido a la cantidad de datos faltantes. Adicionalmente se incluyen pruebas al utilizar las 78 y 169 características mediante el preprocesamiento de la sección 3.3.2.

3.5.1. ZTF Transiente

El conjunto de objetos transientes se encuentra compuesto por 3294 curvas de luz, pertenecientes a 4 clases distintas, las cuales se detallan a continuación. La distribución de cada una de estas se detalla en la Tabla 3.2.

- SNIa: supernova tipo Ia, son causadas por explosiones termonucleares de enanas blancas en sistemas binarios [47]. Este tipo de supernova carece de hidrógeno (H) y tienen una fuerte absorción debida al silicio (Si) cerca de su pico de luminosidad.
- SNII: supernova tipo II, resultan del colapso gravitacional del núcleo de una estrella masiva que ha perdido varias cantidades de su envoltura [47]. Se encuentran compuestas de H.
- SNIbc: supernova tipo Ib ó Ic, se forman debido al colapso del núcleo de una estrella, debido a su propio peso. Se encuentran compuestas de H en el caso de SNIb, y de H y He en el caso de SNIc [47].
- SLSN: supernova super lumínica, es un tipo de explosión estelar con una luminosidad 10 veces superior a una supernova estándar.

Tabla 3.2: Composición de objetos astronómicos del conjunto ZTF Transiente.

Clase	**Cantidad de Curvas**	**Porcentaje [%]**
SNIa	2537	77.0
SNII	546	16.6
SNIbc	160	4.9
SLSN	51	1.5

Además de emplear las curvas de luz, errores de magnitud y marcas de tiempo MJD, se ha optado por utilizar un subconjunto de las 78 características asociadas a la banda g. Esta decisión se basa en la presencia de una gran cantidad valores faltantes o nulos en algunas características (primer método de preprocesamiento de datos). Aunque no todas las características seleccionadas pertenecen al top 30 definido en [42] donde se un modelo *Balanced Random Forest*, durante el entrenamiento del modelo se descartan aquellas que no contribuyen significativamente a la detección de anomalías. Esto ha permitido obtener resultados sólidos, como se evidenciará en el Capítulo 4.

Adicionalmente, se realizaron pruebas utilizando las 78 características de la banda g y todas las 169 características de ambas bandas de observación, siguiendo el segundo método de preprocesamiento de datos. Este enfoque no descarta los valores faltantes o nulos.

La Tabla 3.3 presenta el subconjunto de características de la banda g utilizado mediante el primer método de preprocesamiento de datos, mientras que el resto de las características se detallan más extensamente en el Anexo B.

Tabla 3.3: Subconjunto de características utilizadas para el conjunto de datos ZTF-TRA.

Características			
MHPS_low	n_neg	SPM_tau_fall	Harmonics_phase_2
MHPS_non_zero	n_pos	Psi_CS	Harmonics_phase_3
MHPS_PN_flag	positive_fraction	Psi_eta	Harmonics_phase_4
iqr	n_non_det_before_fid	Harmonics_mag_1	Harmonics_phase_5
delta_mag_fid	n_non_det_after_fid	Harmonics_mag_2	Harmonics_phase_6
delta_mjd_fid	SPM_A	Harmonics_mag_3	Harmonics_phase_7
first_mag	SPM_t0	Harmonics_mag_4	Harmonics_mse
mean_mag	SPM_gamma	Harmonics_mag_5	GP_DRW_sigma
min_mag	SPM_beta	Harmonics_mag_6	GP_DRW_tau
n_det	SPM_tau_rise	Harmonics_mag_7	

3.5.2. ZTF Estocástico

El conjunto de objetos estocásticos se encuentra compuesto por un total de 66992 curvas de luz pertenecientes a 5 clases, las cuales se detallan a continuación. Además, en la Tabla 3.4 se muestra la distribución de datos por clase.

- AGN: nucleo galactico activo (AGN, active galactic nucleus), es una región compacta en el centro de una galaxia que tiene una luminosidad mucho más alta de lo normal.
- QSO: cuásar o quasar (QSO, del inglés quasi-stellar object), es un núcleo galáctico activo, extremadamente luminoso. La emisión de un AGN está alimentada por un agujero negro supermasivo con una masa que oscila entre millones y decenas de miles de millones de masas solares, rodeado por un disco de acreción gaseoso.
- YSO: objeto estelar joven, una estrella que se forma por acumulación de material que cae en una protoestrella desde un disco o envoltura circunestelar.

- Blazar: un blazar es un núcleo galáctico activo (AGN) con un *jet* relativista dirigido muy cerca de un observador (un jet compuesto de materia ionizada que viaja casi a la velocidad de la luz).
- CV/Nova: estrella variable cataclísmica (CV) (inicialmente llamadas nova), son estrellas que aumentan irregularmente su brillo en un factor importante y luego vuelven a caer a un estado de reposo.

Tabla 3.4: Composición de objetos astronómicos del conjunto ZTF Estocástico.

Clase	**Cantidad de Curvas**	**Porcentaje** [%]
QSO	52284	78.0
AGN	7037	10.5
YSO	4757	7.1
Blazar	1774	2.6
CV/Nova	1140	1.8

Al igual que para el conjunto de datos transientes, para el conjunto de datos estocásticos, se utilizó un subconjunto de características (enumeradas en la Tabla 3.4), las cuales fueron seleccionadas debido a que no contaban con una alta cantidad de valores nulos. Adicionalmente se realizaron pruebas con las configuraciones de 78 y 169 características.

Tabla 3.5: Subconjunto de características utilizadas para el conjunto de datos ZTF-EST.

Características			
iqr	positive_fraction	Harmonics_mag_1	Harmonics_phase_4
delta_mag_fid	n_non_det_before_fid	Harmonics_mag_2	Harmonics_phase_5
delta_mjd_fid	n_non_det_after_fid	Harmonics_mag_3	Harmonics_phase_6
first_mag	SPM_A	Harmonics_mag_4	Harmonics_phase_7
mean_mag	SPM_t0	Harmonics_mag_5	Harmonics_mse
min_mag	SPM_gamma	Harmonics_mag_6	GP_DRW_sigma
n_det	SPM_beta	Harmonics_mag_7	GP_DRW_tau
n_neg	SPM_tau_rise	Harmonics_phase_2	
n_pos	SPM_tau_fall	Harmonics_phase_3	

3.5.3. ZTF Periódico

El conjunto de objetos periódicos se encuentra compuesto por un total de 138771 curvas de luz pertenecientes a 6 clases, las cuales se detallan a continuación. Además, en la Tabla 3.6 se muestra la distribución de datos por clase.

- DSCT: estrella pulsante delta scuti o cefeida enana, son un tipo de estrella que muestra variaciones en su luminosidad debido a pulsaciones radiales y no radiales de su superficie.

- LPV: estrella variable de periodo largo, corresponden a estrellas gigantes o supergigantes frías.
- RRL: RR Lyrae, son estrellas pulsantes periódicas, que se encuentran comúnmente en cúmulos globulares.
- CEPH: estrella variable pulsante periódica, pulsa radialmente variando tanto en temperatura como diámetro para producir cambios de brillo con un periodo y amplitud estables muy regulares.
- E: estrella binaria eclipsante, son un par de estrellas que orbitan alrededor de su centro de gravedad común con una orbita inclinada 90 grados con respecto a la Tierra, de modo de que durante su órbita, cada estrella bloquea temporalmente de nuestra vista parte o la totalidad de la luz de la otra estrella.
- Periodic-Other: otros tipos de estrellas variables.

Tabla 3.6: Composición de objetos astronómicos del conjunto ZTF Periódico.

Clase	**Cantidad de Curvas**	**Porcentaje [%]**
DSCT	1134	0.8
LPV	46557	33.5
RRL	41081	29.6
CEPH	896	0.6
E	48231	34.8
Periodic-Other	872	0.6

Al igual que para el conjunto de datos anteriores, para el conjunto de datos periódicos, se utilizó un subconjunto de características (enumeradas en la Tabla 3.6), las cuales fueron seleccionadas debido a que no contaban con una alta cantidad de valores nulos. Adicionalmente se realizaron pruebas con las configuraciones de 78 y 169 características, donde se incluyen los períodos de banda y multibanda, características de gran importancia para la clasificación de objetos periódicos, como se detalla en [42].

Tabla 3.7: Subconjunto de características utilizadas para el conjunto de datos ZTF-PER.

Características			
iqr	Harmonics_phase_3	first_mag	Harmonics_mag_2
SPM_A	Harmonics_phase_4	mean_mag	Harmonics_mag_3
SPM_t0	Harmonics_phase_5	min_mag	Harmonics_mag_4
SPM_gamma	Harmonics_phase_6	n_det	Harmonics_mag_5
SPM_beta	Harmonics_phase_7	n_neg	Harmonics_mag_6
SPM_tau_rise	Harmonics_mse	n_pos	Harmonics_mag_7
SPM_tau_fall	GP_DRW_sigma	positive_fraction	Harmonics_phase_2
GP_DRW_tau	delta_mag_fid	n_non_det_before_fid	
n_non_det_after_fid	delta_mjd_fid	Harmonics_mag_1	

3.6. Entrenamiento de Modelos

Para el entrenamiento de cada modelo utilizado, se sigue un procedimiento consistente. En primer lugar, se divide el conjunto de datos en conjunto de entrenamiento (70 %), validación (10 %) y prueba (20 %). Durante el entrenamiento, solo se utilizan los conjuntos de entrenamiento y validación. Luego, para cada conjunto utilizado, se selecciona una clase como anomalía, y esta se extrae de los conjuntos de entrenamiento, de tal forma que durante el entrenamiento del modelo, este aprenda a partir de solamente las clases restantes del conjunto de datos, las cuales son consideradas como conocidas. De esta manera, el rendimiento de cada modelo en cuanto a la detección de curvas anómalas se mide en que tan bien identifican a la clase que fue removida del conjunto de datos [41].

Adicionalmente, se utiliza el conjunto de validación durante el entrenamiento del modelo SSL, el cual contiene un pequeño porcentaje de datos pertenecientes a la clase removida (30 % *outlier*), con el objetivo de exponer al modelo a la clase desconocida [10] para incrementar las pérdidas del modelo, e incertidumbre, pero sin utilizar en ningún momento las etiquetas de los datos del conjunto de validación, y los datos pertenecientes a este conjunto, para el ajuste de los parámetros del modelo.

Para evaluar el rendimiento del modelo, se consideraron diferentes enfoques en cuanto al equilibrio de los porcentajes de clases *inliers* y *outliers*. En una primera opción, se balancearon los porcentajes para tener alrededor del 50 % de cada clase, lo que permitió evaluar el rendimiento utilizando métricas como AUCPR y AUROC, y así observar la capacidad del modelo para discriminar entre objetos conocidos y desconocidos.

Sin embargo, también se consideró un enfoque más realista donde el conjunto de prueba está compuesto por un mayor porcentaje de clases conocidas (por ejemplo, 90 %) y un porcentaje más reducido de la clase *outlier* (por ejemplo, 10 %) [41]. Esta configuración imita un escenario en el que los eventos anómalos son raros en comparación con los eventos normales. La elección del porcentaje para la evaluación del modelo dependerá de los objetivos específicos del estudio y del contexto de aplicación.

En este trabajo, se optó por realizar principalmente pruebas con el primer escenario donde los porcentajes de clases *inliers* y *outliers* están balanceados de manera similar, es decir 50 % *inlier* y 50 % *outlier*. Esto permite poner a prueba la capacidad del modelo para detectar de manera efectiva las curvas de luz anómalas en un entorno desafiante. Sin embargo, también se realizaron pruebas para el escenario desbalanceado (ver sección 4.4), con el objetivo de verificar la consistencia de los resultados obtenidos por el modelo, y poder realizar comparaciones con el modelo propuesto en [41].

Debido a que no todas las clases cuentan con un distribución de datos equilibrada, para realizar las pruebas con los porcentajes de datos antes mencionados, se tuvieron que utilizar subconjuntos de datos. Estos datos fueron escogidos aleatoriamente de tal forma que si los datos de la clase anómala son menores al 50 % del conjunto de prueba, se quitan datos pertenecientes a las clases conocidas, hasta cumplir el balance de datos deseados. Para el caso contrario, se escoge de forma aleatoria datos pertenecientes a la clase anómala con el objetivo de reducir el porcentaje de datos presentes de esta clase en el conjunto de prueba, hasta alcanzar el balance de datos de prueba.

3.6.1. Selección de Hiper-parámetros y Transformaciones

Como se mencionó anteriormente, se llevó a cabo el entrenamiento del modelo por cada clase de los conjuntos de datos, donde la clase seleccionada se considera como *outlier* y se deja fuera del conjunto de entrenamiento, siendo utilizada dentro del conjunto de prueba y validación. Sin embargo, para obtener los mejores resultados posibles para cada uno de los casos de estudio, se selecciona la mejor configuración de hiperparámetros *HP*, los cuales se listan a continuación:

- Tamaño de entrada (*input size*): el número de características o dimensiones en el conjunto de datos de entrada a la red neuronal. Este valor se determina en función de la dimensión del *embedding* de salida del autoencoder, que es igual al número de características utilizado en cada conjunto de datos.
- Tamaño oculto (*hidden size*): número de neuronas en la capa oculta de la red neuronal.
- Tamaño de las representaciones (*output size*): es la dimensión de las representaciones del espacio latente, es decir, el tamaño de salida del modelo.
- Tamaño de lotes (*batch size*): el número de muestras que se propagarán a través de la red.
- Función de Similitud (*distance metric*): o distancia utilizada como medida de similitud. Puede ser distancia Euclidiana, Coseno, Mahalanobis o modelo KNN con distancia euclidiana.
- Tasa de Aprendizaje (*learning rate*): es un parámetro de ajuste en un algoritmo de optimización que determina el tamaño del paso en cada iteración mientras se mueve hacia un mínimo de una función de pérdida
- Transformaciones (*transform*): transformaciones que son aplicadas de forma aleatoria al conjunto de entrenamiento, con el objetivo de introducir variabilidad e incertidumbre al entrenamiento del modelo. Estas transformaciones se explican con mayor detalle en la sección 3.4.

Con el fin de encontrar la mejor configuración de hiperparámetros para cada modelo, se desarrolló una función que entrena un modelo de aprendizaje auto-supervisado durante 20 épocas para cada una de las configuraciones posibles, con un subconjunto de los datos destinados al entrenamiento del modelo (30 %) con el fin de acelerar la búsqueda de hiperparámetros. Esto debido a que los conjuntos de objetos estocásticos y periódicos cuentan con una gran cantidad de datos. Además, se utilizó un conjunto de validación con proporciones desbalanceadas de clase *inlier* y *outlier* (70 % y 30 % respectivamente) para evaluar el rendimiento del modelo para cada configuración. Los resultados obtenidos mediante esta función se describen en la sección 4.2.

3.7. Evaluación

Como se explicó en la sección correspondiente (sección 3.6), el 20 % del conjunto de datos fue conservado para el conjunto de prueba, el cual contiene la clase que fue seleccionada como anómala, a diferencia del conjunto de entrenamiento. Con el objetivo de evaluar el rendimiento del modelo en un escenario equilibrado, se balancearon los porcentajes de cada clase en el conjunto de prueba, de manera que el 50 % de los datos del conjunto de prueba corresponden a las clases conocidas (*inliers*) y el 50 % restante a la clase desconocida (*outlier*).

En relación a la evaluación del rendimiento del modelo cuando se define una clase como anomalía, se utilizaron métricas como el área bajo la curva de precisión y exhaustividad (AUCPR) y el área bajo la curva característica operativa del Receptor (AUROC), como se detalló en la sección 3.2. Además, para garantizar la robustez y la generalización de los resultados, se utilizaron múltiples semillas. Se realizaron pruebas utilizando 14 semillas diferentes, con el fin de verificar que los puntajes obtenidos por el modelo no se deban a un subconjunto específico de datos seleccionados aleatoriamente. Los resultados obtenidos para cada semilla se promediaron y se calculó la desviación estándar, lo cual se presentará en el próximo capítulo.

Capítulo 4

Resultados y Análisis

4.1. Autoencoder y Uso de Vectores de Características

4.1.1. Entrenamiento Autoencoder

Para el entrenamiento del Autoencoder (AE), se decidió utilizar todas las curvas de luz presentes en los conjuntos de entrenamiento de los datasets ZTF-TRA, ZTF-EST y ZTF-PER en conjunto. Posteriormente, se llevó a cabo un afinamiento del modelo utilizando cada uno de los conjuntos de datos por separado. Esta elección se basó en dos consideraciones principales.

En primer lugar, el objetivo era tener la mayor cantidad de datos posible para entrenar el modelo AE. Al incluir todas las curvas de luz disponibles, se aseguró que el modelo tuviera una amplia variedad de ejemplos para aprender a codificar la información contenida en las curvas de luz. Esto es importante debido a la variabilidad en la cantidad de puntos presentes en las curvas de luz, que van desde un mínimo de 5 observaciones temporales hasta más de 100 puntos. Al proporcionar al modelo una gran cantidad de datos, se facilitó su capacidad para aprender representaciones significativas de las curvas de luz, pero a coste de cierta precisión en las representaciones asociadas a las clases del mismo subconjunto de datos.

En segundo lugar, al entrenar el AE con todas las curvas de luz, se evita que el modelo se centre en aprender a codificar los datos según la clase a la que pertenecen. En los conjuntos de datos astronómicos, puede haber desequilibrios significativos entre las clases, donde algunas clases tienen una cantidad mucho menor de datos en comparación con las clases dominantes.

Si bien es cierto que al entrenar el modelo Autoencoder (AE) con todas las curvas de luz sin hacer distinciones entre clases, existe un sesgo inherente hacia las clases mayoritarias, es importante destacar que al entrenar Autoencoders separados para las curvas transientes, estocásticas y periódicas, este sesgo puede aumentar hacia las clases predominantes, especialmente en conjuntos de datos con una pequeña cantidad de datos como el conjunto ZTF-TRA. Esto se debe a que las codificaciones que aprende el AE están orientadas más hacia las características de las clases mayoritarias, lo que puede afectar la capacidad del modelo para

representar con precisión y detectar las clases minoritarias.

Sin embargo, las curvas de luz entre conjuntos de datos difieren bastante, por lo que puede disminuir el rendimiento del modelo SSL en la asignación de puntajes de anomalía a las curvas de luz. Por esta razón, se lleva a cabo un afinamiento adicional del modelo AE utilizando exclusivamente las curvas de luz pertenecientes al conjunto de datos específico que se está estudiando. Esto posibilita que el modelo generalizado aprenda una representación más significativa de las curvas de luz dentro de ese conjunto particular. Es importante señalar nuevamente que el entrenamiento del modelo AE se centra únicamente en el error de reconstrucción de las curvas de luz y no en la clasificación de clases, por lo que las etiquetas de los datos presentes en el entrenamiento no son tomadas en consideración.

En la Figura 4.1 se presentan las curvas de pérdida del modelo AE durante el proceso de entrenamiento, tanto para el conjunto de entrenamiento como para el conjunto de validación. Se observa que el modelo logra converger a una solución estable alrededor de las 15 épocas de entrenamiento.

Este resultado puede atribuirse a la arquitectura del Autoencoder (AE), que se detalla en el Anexo C.1. Esta arquitectura incluye dos capas convolucionales que desempeñan un papel fundamental al extraer características iniciales de las curvas de luz antes de pasar a las capas LSTM. Estas capas convolucionales permiten al modelo capturar patrones en las curvas de luz a un nivel más detallado y extraer características relevantes para el aprendizaje.

Además, el uso de un tamaño de lote (batch size) grande, que comprende aproximadamente 30,000 objetos astronómicos, también contribuye a la rápida convergencia del modelo. Esto se debe a que el tamaño de lote grande permite procesar una gran cantidad de información en cada iteración, lo que a su vez evita lotes con una cantidad limitada de puntos de datos que podrían ralentizar la convergencia del modelo.

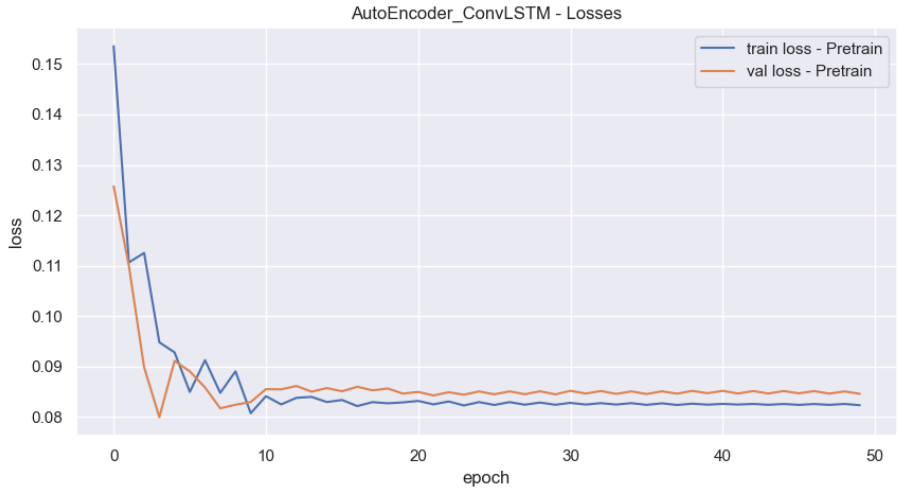
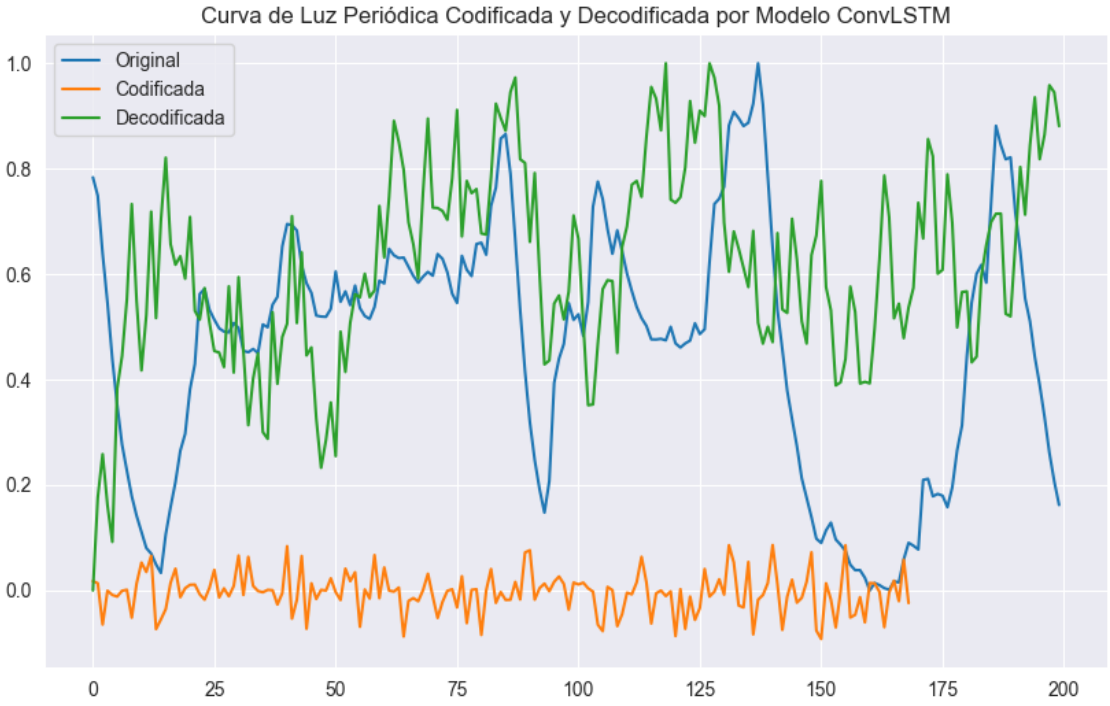
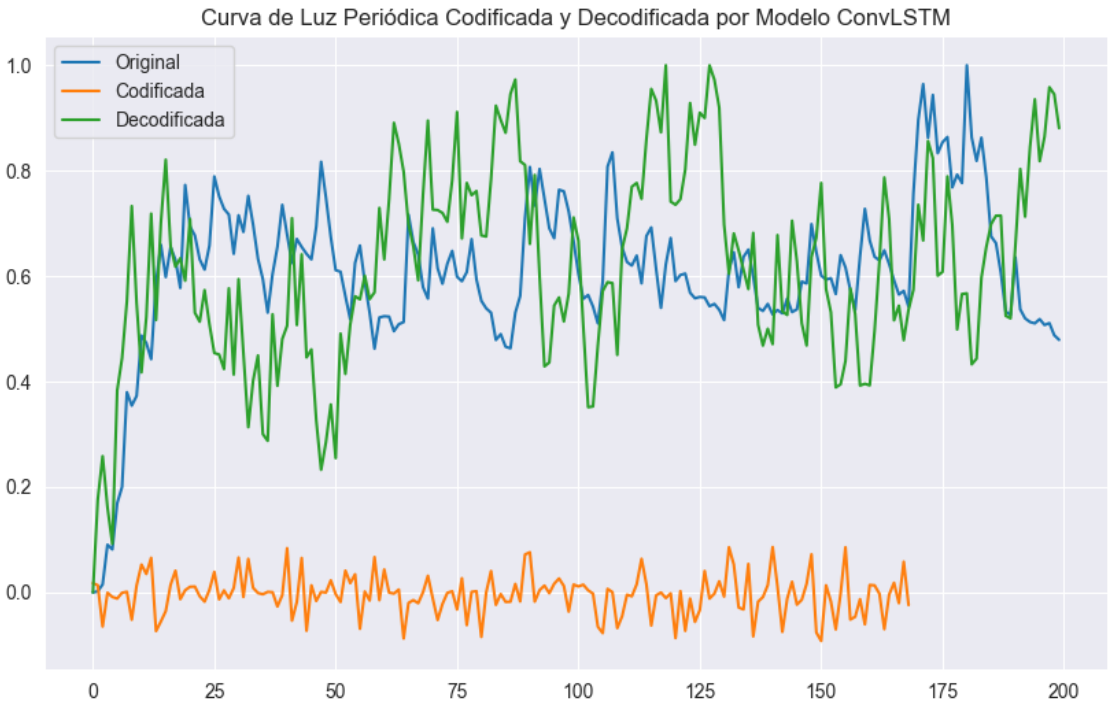


Figura 4.1: Curva de pérdidas del modelo AE durante el proceso de entrenamiento, para los conjuntos de entrenamiento y de validación.



(a) Muestra 1: Curva de luz de objeto astronómico periódico.



(b) Muestra 2: Curva de luz de objeto astronómico periódico.

Figura 4.2: Ejemplo de codificación y decodificación de una curva de luz por medio del modelo AE.

En la Figura 4.2 se muestran dos ejemplos de codificación y decodificación de una curva de luz por medio del modelo AE. Se observa que, para ambos ejemplos, aunque existen errores o diferencias en la versión decodificada en comparación con la curva original, las fluctuaciones en la curva se mantienen notablemente similares. Esto es especialmente significativo dada la desafiante tarea que el modelo AE aprende a realizar: codificar curvas de luz de tamaños irregulares y períodos diversos para cada clase y objeto astronómico.

4.1.2. Uso de Autoencoder

En las Tablas 4.1 y 4.2, se presentan los resultados del modelo MLP entrenado mediante aprendizaje auto-supervisado para las métricas AUCPR y AUROC, respectivamente. Estas tablas incluyen los resultados tanto utilizando el Autoencoder (AE) para la codificación de las curvas de luz (SSL) como sin utilizarlo (SSL-nAE). Además, se incluyen los resultados obtenidos al no utilizar el AE para los conjuntos de datos transientes, estocásticos y periódicos en las Tablas 4.4 a 4.11.

Los resultados indican que en la mayoría de los casos, la incorporación del Autoencoder conlleva una mejora significativa, especialmente en la identificación de la clase LPV. Sin embargo, para las clases en las que los resultados con el Autoencoder son inferiores, las diferencias en la detección de curvas anómalas no son sustanciales, ya que los resultados tienden a estar ligeramente por debajo de un resultado aleatorio según la métrica AUCPR y cercanos al azar en cuanto a la métrica AUROC.

Esta tendencia también se logra observar para los conjuntos de datos transientes y estocásticos, donde para algunas clases la diferencia es aún mayor que las obtenidas para el caso periódico. Por esta razón, se decidió continuar los experimentos haciendo uso del Autoencoder entrenado con todas las curvas de luz.

4.1.3. Uso de Vector de Características

En la sección 3.5, se presentan subconjuntos de características específicas relacionadas con cada objeto astronómico en la banda de observación g [42] [48]. Para analizar el impacto de estas características en el entrenamiento del modelo de aprendizaje auto-supervisado, se llevaron a cabo pruebas en las que el modelo SSL se entrenó sin incorporar el vector de características en un cuarto canal adicional (modelo SSL-nFeats), y pruebas que solamente utilizan el vector de características (modelo SSL-nAE-AllFeats). Posteriormente, se realizaron predicciones en el conjunto de prueba. Los resultados de estas pruebas se resumen en las Tablas 4.1 y 4.2 para el conjunto de datos ZTF-PER, y en las Tablas 4.4 a 4.11 para los demás conjuntos de datos.

Para ambas métricas, podemos observar que los resultados obtenidos por el modelo cuando no utiliza el vector de características son prácticamente equivalentes a un escenario aleatorio, aproximadamente alrededor del 50 %. Esto se debe a que el modelo está diseñado para discernir si un objeto astronómico es conocido o no, lo que se traduce en dos clases, lo que resulta en un valor del 50 % para AUROC. En el caso de AUCPR, el valor aleatorio se determina

en función del equilibrio de las clases, y como ambas clases están igualmente equilibradas, el valor aleatorio es el mismo que para AUROC, es decir, el 50 %.

A partir de ambas métricas, podemos observar una mejora significativa al incorporar el vector de características en conjunto con el Autoencoder (modelo SSL). Si bien, los resultados obtenidos para las clases CEPH y E son mayores al utilizar solamente el vector de características, la diferencia no es significativa en comparación al resto de clases del conjunto ZTF-PER.

Además, en la sección 4.3 se describen resultados obtenidos por el modelo SSL en el resto de conjunto de datos, donde la diferencia entre los resultados de AUROC y AUCPR es significativa. Por lo tanto, se tomó la decisión de incluir el vector de características como un canal adicional durante el entrenamiento del modelo propuesto (SSL).

Tabla 4.1: Comparación del modelo SSL propuesto al utilizar el modelo AE para la codificación de curvas de luz, frente al modelo SSL sin el uso del modelo AE, el modelo SSL sin utilizar el vector de características asociadas, y el modelo SSL utilizando solamente el vector de características con respecto a la métrica AUCPR en el conjunto de datos ZTF-PER. Los resultados mostrado para cada clase corresponden al caso donde la respectiva clase es seleccionada como *outlier*.

AUCPR	Periódico					
Modelo	DSCT	P-Other	RRL	LPV	CEPH	E
SSL	0.491 ± 0.016	**0.611 ± 0.026**	**0.606 ± 0.015**	**0.786 ± 0.001**	0.525 ± 0.021	0.481 ± 0.002
SSL-nAE	0.496 ± 0.016	0.586 ± 0.026	0.515 ± 0.003	0.583 ± 0.002	0.511 ± 0.019	0.499 ± 0.002
SSL-nFeats	0.502 ± 0.026	0.512 ± 0.027	0.502 ± 0.004	0.500 ± 0.004	0.512 ± 0.031	0.499 ± 0.003
SSL-nAE-AllFeats	**0.567 ± 0.022**	0.574 ± 0.033	0.521 ± 0.004	0.548 ± 0.003	**0.546 ± 0.027**	**0.534 ± 0.002**

Tabla 4.2: Comparación del modelo SSL propuesto al utilizar el modelo AE para la codificación de curvas de luz, frente al modelo SSL sin el uso del modelo AE, y el modelo SSL sin utilizar el vector de características asociadas, y el modelo SSL utilizando solamente el vector de características con respecto a la métrica AUROC en el conjunto de datos ZTF-PER. Los resultados mostrado para cada clase corresponden al caso donde la respectiva clase es seleccionada como *outlier*.

AUROC	Periódico					
Modelo	DSCT	P-Other	RRL	LPV	CEPH	E
SSL	0.513 ± 0.023	**0.654 ± 0.027**	**0.618 ± 0.018**	**0.759 ± 0.002**	0.527 ± 0.020	0.526 ± 0.003
SSL-nAE	0.496 ± 0.022	0.615 ± 0.023	0.530 ± 0.003	0.573 ± 0.003	0.500 ± 0.022	0.496 ± 0.003
SSL-nFeats	0.491 ± 0.028	0.510 ± 0.029	0.502 ± 0.005	0.499 ± 0.006	0.502 ± 0.037	0.499 ± 0.004
SSL-nAE-AllFeats	**0.567 ± 0.022**	0.574 ± 0.033	0.521 ± 0.004	0.548 ± 0.003	**0.546 ± 0.027**	**0.534 ± 0.002**

4.2. Selección de Hiperparámetros

En la sección 3.6.1, se hizo mención al desarrollo de una función que realiza pruebas con diversas configuraciones de hiperparámetros con el fin de maximizar los resultados para cada modelo en la detección de objetos astronómicos desconocidos. Para agilizar el proceso de búsqueda de hiperparámetros, se utilizó un subconjunto del conjunto de entrenamiento (30 %) debido a que algunas clases contenían una gran cantidad de objetos astronómicos. Además, se utilizó el conjunto de validación que conserva los datos asociados a la clase *outlier*, permitiendo así evaluar el rendimiento del modelo para cada combinación posible.

Los resultados obtenidos a través de la función implementada se presentan en detalle en la Tabla 4.3. En la columna *clase* se indica la clase del conjunto de datos seleccionada como anomalía, y en las columnas adyacentes se muestran los valores óptimos de los hiperparámetros encontrados por la función, los cuales maximizan el rendimiento del modelo. Cada uno de los hiperparámetros utilizados se describen en la sección 3.6.1, junto a las transformaciones utilizadas (sección 3.4).

Como se logra observar en los resultados de la Tabla 4.3, solamente se obtuvo que las transformaciones Deformación del Tiempo (*time warping*), Deformación de Magnitud (*magnitude warping*) y Escalamiento Aleatorio (*random scaling*) mejoran el rendimiento del modelo. Esto último se puede deber a que el resto de transformaciones puede introducir errores en las clases a las que pertenecen los objetos, ya que al realizar la Inversión Temporal de la curva de luz, el objeto puede cambiar su clase apareciendo un periodo que en verdad no existe, o simplemente no aportan con la suficiente variabilidad a la curva original para que se pueda diferenciar de la clase correspondiente, e incrementar el rendimiento del modelo.

Una vez determinados los valores de hiperparámetros que maximizan el rendimiento del modelo SSL, se procedió a entrenar un modelo para cada clase durante un máximo de 50 épocas, siguiendo el procedimiento explicado en el capítulo anterior, es decir, se entrena un modelo MLP mediante aprendizaje auto-supervisado por cada clase presente en el conjunto de datos. Esta clase se retira del conjunto de entrenamiento, pero se conserva un porcentaje del 30 % en el conjunto de validación, para exponer al modelo a clases desconocidas, pero sin realizar la retroalimentación del modelo con las curvas presentes en el conjunto de validación y sin utilizar las etiquetas de los datos, como en el trabajo realizado en [10]. Posteriormente, se calcularon los puntajes de anomalía para cada objeto del conjunto de prueba, con el objetivo de evaluar el rendimiento del modelo según las métricas detalladas en secciones anteriores. Estos resultados se muestran en la sección 4.3.

Cabe destacar que las dimensiones especificadas en la Tabla 4.3 corresponden a la dimensión del vector de características al utilizar un subconjunto de las caracteristicas de la banda g, por lo que para los casos donde se utilizan 78 y 169 características, el input size es igual a 78 y 169 respectivamente.

Tabla 4.3: Mejores combinaciones de hiperparámetros encontradas para cada una de las clases considerada anómala de los conjuntos ZTF-TRA, ZTF-EST y ZTF-PER.

Clase Outlier	Input Size	Hidden Size	Output Size	Learning Rate	Transform	Distance Metric
SLSN	39	32	15	0.0001	Time Warping	Mahalanobis
SNII	39	32	15	0.001	Time Warping	Mahalanobis
SNIa	39	64	10	0.001	Time Warping	Coseno
SNIbc	39	32	15	0.001	Time Warping	KNN
AGN	34	32	10	0.001	Random Scaling	KNN
BLAZAR	34	64	10	0.001	Time Warping	KNN
CV/NOVA	34	32	15	0.001	Time Warping	KNN
QSO	34	32	15	0.001	Time Warping	Coseno
YSO	34	32	10	0.001	Random Scaling	KNN
DSCT	34	32	10	0.0001	Magnitude Warping	Mahalanobis
RRL	34	64	10	0.001	Time Warping	Coseno
LPV	34	64	15	0.0001	Time Warping	KNN
CEPH	34	32	10	0.001	Random Scaling	KNN
E	34	64	15	0.0001	Time Warping	Coseno
P-Other	34	32	15	0.0001	Magnitude Warping	KNN

4.3. Evaluación de Modelos

4.3.1. ZTF Transiente

Las Tablas 4.4 y 4.5 muestran los resultados obtenidos por el modelo de aprendizaje auto-supervisado propuesto, en comparación con los modelos basados en características, para cada una de las clases del conjunto de datos de objetos Transientes, evaluados según las métricas AUCPR y AUROC, respectivamente. Se puede observar que el modelo propuesto supera a todos los modelos basados en características en términos de ambas métricas de evaluación.

Al analizar los resultados para la clase SLSN, se observa que el modelo propuesto logra discriminar en promedio entre las clases *inliers* y *outlier* para una gran cantidad de objetos en el conjunto de prueba. Sin embargo, al examinar la desviación de los resultados obtenidos para cada semilla utilizada, tanto en AUCPR como en AUROC, se identifica cierta variabilidad. Se observa que en algunos subconjuntos, el modelo obtiene un área bajo la curva significativamente mayor al promedio (alrededor de 0.8), lo que indica que, a pesar de tener en promedio un valor alto de área bajo la curva, el modelo no es tan consistente en la tarea de discriminación de esta clase como se esperaría.

En lo que respecta a las demás clases presentes en el conjunto de datos, es notable que todas ellas obtienen valores superiores a 0.6 para ambas métricas que se han empleado. Esto indica que el modelo propuesto demuestra ser eficaz en la detección de objetos anómalos en estas clases. Sin embargo, es importante señalar que al utilizar las 78 y 169 características derivadas de la banda de observación g y de las bandas g y r en conjunto (modelos SSL-allFeats-g y SSL-allFeats-gr respectivamente), los resultados muestran una disminución considerable en comparación con el modelo SSL base. Por lo tanto, parece más beneficioso

emplear el subconjunto de características utilizado en el modelo SSL base para obtener mejores resultados.

Si observamos los resultados obtenidos por los modelos basados en caracteristicas (SVM, ISO y LOF), los cuales utilizan las 169 características para cada objeto astronómico y no hacen uso de las curvas de luz como sí lo hacen los modelos basados en redes neuronales, se puede observar que tanto el modelo ISO como el modelo LOF obtienen resultados bastante similares, y que rondan en 0.6, es decir, un poco por sobre un resultado aleatorio, a diferencia del modelo SVM que se encuentra un poco por debajo de los modelos de aprendizaje autosupervisado. Esto se puede deber a que la distribución de los datos se encuentran en regiones más aisladas, permitiendo que el modelo SVM pueda diferenciarlas de mejor manera, a diferencia del modelo LOF que se basa en densidad, y el modelo ISO que construye arboles de decision para aislar anomalías, que se pueden ver afectados por la cantidad de caracteristicas que no son altamente informativas.

A partir de los resultados del test estadístico de permutación (ver Tabla 4.6), se puede decir que el modelo propuesto supera a todos los modelos basados en características, ya que no solo obtiene una buena precisión sobre la clase *outlier* en la mayoría de clases, sino que la diferencia es sustancial, al obtener un valor p inferior a 0.05 que rechaza la hipotesis nula H_0. De esta misma manera, se puede decir que los resultados obtenidos por el modelo SVM para la clase SNIbc, al ser estadísticamente diferencables, son superiores al modelo SSL.

Tabla 4.4: Comparación del modelo SSL propuesto con modelos basados en métricas para el conjunto de datos ZTF-TRA, mediante la métrica de evaluación AUCPR, considerando la clase *outlier* como clase positiva.

AUCPR	Transiente			
Modelo	SLSN	SNII	SNIa	SNIbc
SSL	**0.723 ± 0.119**	**0.696 ± 0.032**	**0.683 ± 0.024**	0.633 ± 0.050
SSL-allFeats-g	0.672 ± 0.126	0.616 ± 0.037	0.520 ± 0.020	0.586 ± 0.073
SSL-allFeats-gr	0.671 ± 0.081	0.635 ± 0.049	0.575 ± 0.027	0.518 ± 0.040
SSL-nAE	0.641 ± 0.091	0.624 ± 0.027	0.504 ± 0.021	**0.642 ± 0.054**
SSL-nFeats	0.571 ± 0.070	0.507 ± 0.041	0.522 ± 0.022	0.539 ± 0.050
TF-C	0.570 ± 0.105	0.504 ± 0.026	0.537 ± 0.022	0.544 ± 0.050
TS-TCC	0.580 ± 0.085	0.517 ± 0.026	0.512 ± 0.030	0.534 ± 0.065
SVM*	0.667 ± 0.236	0.533 ± 0.038	0.579 ± 0.049	0.639 ± 0.128
ISO*	0.550 ± 0.163	0.503 ± 0.010	0.527 ± 0.020	0.556 ± 0.049
LOF*	0.533 ± 0.194	0.492 ± 0.011	0.504 ± 0.005	0.517 ± 0.041

Tabla 4.5: Comparación del modelo SSL propuesto con modelos basados en métricas para el conjunto de datos ZTF-TRA, mediante la métrica de evaluación AUROC, considerando la clase *outlier* como clase positiva.

AUROC	Transiente			
Modelo	SLSN	SNII	SNIa	SNIbc
SSL	**0.703 ± 0.103**	**0.701 ± 0.029**	**0.677 ± 0.019**	0.617 ± 0.046
SSL-allFeats-g	0.596 ± 0.159	0.609 ± 0.043	0.526 ± 0.031	0.570 ± 0.093
SSL-allFeats-gr	0.625 ± 0.106	0.642 ± 0.030	0.571 ± 0.036	0.524 ± 0.055
SSL-nAE	0.642 ± 0.094	0.613 ± 0.022	0.491 ± 0.021	0.623 ± 0.045
SSL-nFeats	0.513 ± 0.094	0.490 ± 0.046	0.513 ± 0.026	0.510 ± 0.058
TF-C	0.505 ± 0.134	0.486 ± 0.030	0.529 ± 0.022	0.526 ± 0.056
TS-TCC	0.529 ± 0.097	0.511 ± 0.036	0.499 ± 0.037	0.518 ± 0.081
SVM*	0.650 ± 0.200	0.556 ± 0.063	0.600 ± 0.059	**0.650 ± 0.161**
ISO*	0.550 ± 0.187	0.506 ± 0.021	0.550 ± 0.036	0.595 ± 0.062
LOF*	0.550 ± 0.100	0.483 ± 0.022	0.508 ± 0.009	0.531 ± 0.038

Tabla 4.6: Valores p de cada modelo de referencia con respecto al modelo propuesto SSL, en el conjunto de datos Transientes (ZTF-TRA).

Clase Outlier	$P_{SSL-TFC}$	$P_{SSL-TCC}$	$P_{SSL-ISO}$	$P_{SSL-LOF}$	$P_{SSL-SVM}$	$P_{SSL-nAE}$	$P_{SSL-nFeats}$	$P_{SSL-allFeats}$
SNIa	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$
SNII	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$
SNIbc	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$
SLSN	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$

4.3.2. ZTF Estocástico

En las Tablas 4.7 y 4.8, se presentan los resultados obtenidos por los modelos utilizados para el conjunto de datos de objetos estocásticos, al ser evaluados con las métricas AUCPR y AUROC, respectivamente. Al igual que en el caso de los objetos transientes, el modelo de aprendizaje auto-supervisado muestra resultados superiores en la mayoría de las clases de los objetos estocásticos. Sin embargo, se observa que las clases QSO y AGN obtuvieron los peores resultados tanto en AUCPR como en AUROC. De hecho, los resultados para la clase QSO fueron apenas superiores a los obtenidos por los modelos basados en características. Esta tendencia también se observa para el caso donde se utilizan las 78 características de la banda g para el entrenamiento del modelo, pero los resultados obtenidos se encuentran por debajo a los del modelo al utilizar el subconjunto de características. Sin embargo, al utilzar todas las características de los objetos estocásticos, el modelo de aprendizaje auto-supervisado muestra resultados similares, pero con un gran incremento para la clase QSO, por lo que ciertas características multibanda pueden ser bastante relevantes para esta clase.

Si consideramos los porcentajes de objetos astronómicos que conforman cada clase dentro del conjunto de datos estocásticos, podemos notar que las clases QSO y AGN contienen la gran mayoría de curvas de luz. Es posible que el rendimiento del modelo basado en aprendizaje auto-supervisado se vea afectado directamente por la cantidad de elementos presentes en el conjunto de entrenamiento. Dado que este tipo de modelos se entrenan sin utilizar etiquetas,

deben aprender a identificar características o patrones en los datos que les permitan distinguir si un objeto astronómico pertenece a una clase conocida o si es una anomalía.

Es comprensible que las clases QSO y AGN obtengan resultados más pobres, ya que el modelo puede enfrentar dificultades para aprender características distintivas debido a la gran cantidad de datos que aportan estas clases, y el enfoque contrastivo que tiene el entrenamiento del modelo. Por otro lado, es esperable que el modelo tenga un mejor desempeño en las clases como YSO, BLAZAR y CV/NOVA, donde los resultados son buenos. Esto se debe a que el modelo logra aprender características relevantes de los datos proporcionados por las clases restantes. Sin embargo, cuando se trata de las clases que conforman la gran mayoría de los datos presentes en el conjunto de datos, como objetos desconocidos, las características aprendidas por el modelo pueden no ser lo suficientemente discriminativas para distinguir entre lo conocido y lo desconocido.

Es importante tener en cuenta que el enfoque utilizado en este estudio consistió en evaluar una a una las clases del conjunto de datos para demostrar la capacidad del modelo de aprendizaje auto-supervisado en la detección de curvas de luz anómalas o desconocidas. Sin embargo, esto no implica necesariamente que sea el método óptimo para el entrenamiento del modelo, ya que los resultados obtenidos indican que los modelos de aprendizaje auto-supervisado requieren una gran cantidad de datos y un cierto nivel de balance en la distribución de las clases para lograr un rendimiento óptimo.

Por lo tanto, para obtener mejores resultados en la detección de curvas de luz anómalas o desconocidas, es necesario contar con una gran cantidad de datos y una distribución balanceada de las clases en el conjunto de entrenamiento. Esto permitirá que el modelo de aprendizaje auto-supervisado aprenda características distintivas de manera más efectiva y logre discriminar de manera más precisa entre objetos conocidos y anomalías. Además, se puede estudiar el efecto de incluir características al utilizar colores obtenidos de AllWISE como en [42], ya que el modelo que utiliza las 169 características incluye colores, pero estos pueden no ser los mejores para la detección de anomalías.

Tabla 4.7: Comparación del modelo SSL propuesto con modelos basados en métricas para el conjunto de datos ZTF-EST, mediante la métrica de evaluación AUCPR, considerando la clase *outlier* como clase positiva.

AUCPR	Estocástico				
Modelo	AGN	Blazar	CV/Nova	QSO	YSO
SSL	**0.592 ± 0.010**	0.703 ± 0.018	**0.954 ± 0.007**	0.558 ± 0.013	**0.859 ± 0.010**
SSL-allFeats-g	0.573 ± 0.009	0.642 ± 0.018	0.947 ± 0.014	0.562 ± 0.007	0.626 ± 0.014
SSL-allFeats-gr	0.544 ± 0.010	**0.712 ± 0.015**	0.888 ± 0.021	**0.761 ± 0.005**	0.733 ± 0.010
SSL-nAE	0.509 ± 0.006	0.573 ± 0.013	0.762 ± 0.018	0.495 ± 0.004	0.690 ± 0.011
SSL-nFeats	0.502 ± 0.008	0.519 ± 0.021	0.506 ± 0.026	0.500 ± 0.008	0.503 ± 0.011
TF-C	0.507 ± 0.006	0.511 ± 0.013	0.516 ± 0.021	0.504 ± 0.007	0.499 ± 0.015
TS-TCC	0.503 ± 0.013	0.509 ± 0.020	0.514 ± 0.031	0.502 ± 0.007	0.502 ± 0.014
SVM*	0.564 ± 0.043	0.556 ± 0.006	0.687 ± 0.033	0.511 ± 0.003	0.631 ± 0.018
ISO*	0.503 ± 0.002	0.497 ± 0.005	0.545 ± 0.020	0.512 ± 0.004	0.502 ± 0.003
LOF*	0.500 ± 0.002	0.495 ± 0.006	0.494 ± 0.009	0.526 ± 0.005	0.497 ± 0.003

Tabla 4.8: Comparación del modelo SSL propuesto con modelos basados en métricas para el conjunto de datos ZTF-EST, mediante la métrica de evaluación AUROC, considerando la clase *outlier* como clase positiva.

AUROC	Estocástico				
Modelo	AGN	Blazar	CV/Nova	QSO	YSO
SSL	**0.628 ± 0.008**	**0.721 ± 0.015**	**0.955 ± 0.004**	0.561 ± 0.021	**0.867 ± 0.008**
SSL-allFeats-g	0.557 ± 0.008	0.657 ± 0.020	0.947 ± 0.009	0.557 ± 0.007	0.634 ± 0.018
SSL-allFeats-gr	0.568 ± 0.009	0.714 ± 0.017	0.902 ± 0.013	**0.787 ± 0.004**	0.738 ± 0.007
SSL-nAE	0.504 ± 0.008	0.581 ± 0.012	0.721 ± 0.020	0.495 ± 0.005	0.685 ± 0.007
SSL-nFeats	0.502 ± 0.008	0.515 ± 0.023	0.499 ± 0.022	0.500 ± 0.010	0.497 ± 0.010
TF-C	0.507 ± 0.008	0.502 ± 0.016	0.499 ± 0.031	0.505 ± 0.010	0.497 ± 0.015
TS-TCC	0.503 ± 0.012	0.507 ± 0.026	0.505 ± 0.025	0.500 ± 0.008	0.500 ± 0.016
SVM*	0.603 ± 0.067	0.556 ± 0.006	0.688 ± 0.030	0.522 ± 0.005	0.635 ± 0.017
ISO*	0.507 ± 0.005	0.494 ± 0.009	0.580 ± 0.037	0.524 ± 0.008	0.503 ± 0.005
LOF*	0.499 ± 0.003	0.490 ± 0.011	0.488 ± 0.013	0.549 ± 0.008	0.494 ± 0.004

A partir de los resultados del test estadístico de permutación de la Tabla 4.9, se puede decir que los resultados obtenidos por el modelo propuesto son superiores a los obtenidos por el resto de modelos, ya que no solo obtiene una buena precisión sobre la clase *outlier* en la mayoría de clases, sino que la diferencia es sustancial, al obtener un valor p inferior a 0.05 que rechaza la hipotesis nula H_0 (ver sección 3.1.4).

Tabla 4.9: Valores p de cada modelo de referencia con respecto al modelo propuesto SSL, en el conjunto de datos Estocásticos (ZTF-EST).

Clase Outlier	$P_{SSL-TFC}$	$P_{SSL-TCC}$	$P_{SSL-ISO}$	$P_{SSL-LOF}$	$P_{SSL-SVM}$	$P_{SSL-nAE}$	$P_{SSL-nFeats}$	$P_{SSL-allFeats}$
QSO	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$
AGN	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$
YSO	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$
BLAZAR	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$
CV/Nova	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$2 \cdot 10^{-2}$	$3 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$

4.3.3. ZTF Periódico

Si analizamos los resultados obtenidos por el modelo SSL propuesto (Tablas 4.10 y 4.11) en comparación con los modelos basados en características, podemos observar que en general los resultados son superiores, con la excepción de las clase DSCT, CEPH y E en el conjunto de datos Periódicos. Sin embargo, al utilizar las 78 características de la banda g (modelo SSL-allFeats), los resultados para estas clases mejoran, obteniendo mejores resultados que el resto de modelos. A la vez, al utilizar todas las características de la banda g, banda r y multibanda (modelo SSL-allFeats-gr), los resultados para estas clases mejoran aún más, especialmente para las clases CEPH y E, obteniendo mejores resultados que el resto de modelos. Esto se puede deber al hecho de que ciertos atributos como el periodo multibanda son bastante importantes para la identificación de los objetos periódicos [42].

A diferencia del caso anterior, el conjunto de curvas estocásticas cuenta con 3 clases mayoritarias (E, RRL y LPV), donde las clases RRL y LPV están más equilibradas entre sí en términos de proporción de datos, a diferencia de los casos anteriores donde se tenía un mayor desbalance entre clases. Sin embargo, estas clases constituyen casi la totalidad de los datos presentes en el conjunto de datos, y a diferencia del conjunto transiente, cuenta con alrededor de 48 veces más datos, lo que explica por qué obtienen mejores resultados en comparación con las clases minoritarias.

El principal desafío se presenta al tratar de identificar las clases DSCT y E. Sin embargo, cuando se consideran todas las características en lugar de codificar las curvas de luz en un espacio latente de menor dimensión, se observa una mejora en los resultados para la mayoría de las clases. Además, se nota una tendencia a obtener mejores resultados para las clases predominantes en comparación con las demás clases, a diferencia de lo observado en los conjuntos anteriores.

Es posible que al codificar las curvas de luz en un espacio latente de menor dimensión, la red neuronal esté perdiendo información crucial de estas clases, lo que limita la capacidad del modelo para aprender patrones esenciales, como la identificación del período de los objetos en el conjunto de entrenamiento. Como resultado, el modelo no logra identificar de manera más efectiva los objetos de estas clases en comparación con un enfoque aleatorio.

Es importante destacar que estos resultados indican que el modelo propuesto tiene un buen desempeño en la detección de curvas de luz anómalas o desconocidas en las clases mayoritarias, pero enfrenta dificultades para identificar patrones distintivos en las clases minoritarias debido a la falta de datos y la posible pérdida de información en la codificación en el espacio latente. Este hallazgo resalta la importancia de contar con un conjunto de datos equilibrado y suficiente para todas las clases con el fin de lograr un rendimiento óptimo en la detección de anomalías.

En el caso de las clases minoritarias, como DSCT en el conjunto de datos Periódicos, la consideración de todas estas clases como *outliers* permitiría al modelo enfocarse específicamente en identificar patrones distintivos de estas clases, sin verse afectado por la presencia abrumadora de las clases mayoritarias (como el trabajo realizado en [4]). Al no tener que competir las clases mayoritarias con las minoritarias durante el entrenamiento, el modelo tendría más capacidad para aprender características específicas de las clases y, en consecuencia, mejorar su capacidad de detección.

Este enfoque también podría ayudar a abordar el desafío de la pérdida de información en la codificación de curvas de luz en un espacio latente de menor dimensión, ya que en el presente trabajo se realizaron pruebas con varios largos de codificación, los cuales en determinados casos no fueron suficiente para garantizar la detección de anomalías. De esta manera, al considerar todas las clases minoritarias como *outliers*, se le otorgaría al modelo la oportunidad de capturar de manera más efectiva las peculiaridades y patrones únicos presentes en estas clases que no fueron posibles de codificar al utilizar dimensiones más grandes, lo que podría resultar en una mejora significativa en la detección de anomalías.

En cuanto a los resultados de los otros modelos basados en aprendizaje auto-supervisado, se puede observar que estos no obtuvieron mejores resultados que un modelo puramente aleatorio. Esto se puede deber a que estos modelos fueron diseñados con una fuerte dependencia temporal, con datos muestreados regularmente como es el caso de las señales EEG, por lo que al enfrentarse a series temporales muestreadas irregularmente como las curvas de luz utilizadas en el presente trabajo, los modelos no son capaces de aprender patrones esenciales.

Si examinamos los resultados obtenidos por los modelos basados en características (SVM, ISO y LOF) para cada clase seleccionada como *outlier*, se puede observar que no logran alcanzar resultados muy prometedores. Esto podría atribuirse al hecho de que, en algunos casos, los datos no presentan regiones lo suficientemente aisladas como para que el modelo pueda distinguirlas de manera efectiva, especialmente en el caso del modelo SVM. Esto contrasta con el enfoque del modelo LOF, que se basa en la densidad, y el modelo ISO, que construye árboles de decisión para aislar anomalías. Estos últimos modelos podrían verse afectados por la presencia de características que no son altamente informativas.

Tabla 4.10: Comparación del modelo SSL propuesto con modelos basados en métricas para el conjunto de datos ZTF-PER, mediante la métrica de evaluación AUCPR, considerando la clase *outlier* como clase positiva.

AUCPR	Periódico					
Modelo	DSCT	P-Other	RRL	LPV	CEPH	E
SSL	0.491 ± 0.016	**0.611 ± 0.026**	0.606 ± 0.015	0.786 ± 0.001	0.525 ± 0.021	0.481 ± 0.002
SSL-allFeats-g	**0.556 ± 0.041**	0.559 ± 0.051	0.580 ± 0.004	0.816 ± 0.002	0.557 ± 0.020	0.637 ± 0.003
SSL-allFeats-gr	0.534 ± 0.024	0.520 ± 0.023	**0.797 ± 0.003**	**0.978 ± 0.000**	**0.680 ± 0.033**	**0.746 ± 0.004**
SSL-nAE	0.496 ± 0.016	0.586 ± 0.026	0.515 ± 0.003	0.583 ± 0.002	0.511 ± 0.019	0.499 ± 0.002
SSL-nFeats	0.502 ± 0.026	0.512 ± 0.027	0.502 ± 0.004	0.500 ± 0.004	0.512 ± 0.031	0.499 ± 0.003
TF-C	0.515 ± 0.025	0.516 ± 0.028	0.502 ± 0.004	0.501 ± 0.005	0.510 ± 0.026	0.500 ± 0.004
TS-TCC	0.509 ± 0.022	0.513 ± 0.024	0.500 ± 0.004	0.501 ± 0.003	0.502 ± 0.024	0.500 ± 0.003
SVM*	0.533 ± 0.023	0.489 ± 0.036	0.525 ± 0.004	0.575 ± 0.028	0.540 ± 0.042	0.504 ± 0.002
ISO*	0.536 ± 0.019	0.510 ± 0.011	0.527 ± 0.003	0.498 ± 0.000	0.545 ± 0.011	0.525 ± 0.003
LOF*	0.502 ± 0.003	0.500 ± 0.010	0.501 ± 0.001	0.499 ± 0.001	0.501 ± 0.008	0.502 ± 0.001

Al igual que para los conjuntos de datos anteriores, a partir de los resultados del test estadístico de permutación de la Tabla 4.12, se puede decir que los resultados obtenidos por el modelo propuesto son superiores a los obtenidos por el resto de modelos, ya que no solo obtiene una buena precisión sobre la clase *outlier* en la mayoría de clases, sino que la diferencia es sustancial, al obtener un valor p inferior a 0.05 que rechaza la hipotesis nula H_0.

Tabla 4.11: Comparación del modelo SSL propuesto con modelos basados en métricas para el conjunto de datos ZTF-PER, mediante la métrica de evaluación AUROC, considerando la clase *outlier* como clase positiva.

AUROC	Periódico					
Modelo	DSCT	P-Other	RRL	LPV	CEPH	E
SSL	0.513 ± 0.023	**0.654 ± 0.027**	0.618 ± 0.018	0.759 ± 0.002	0.527 ± 0.020	0.526 ± 0.003
SSL-allFeats-g	**0.576 ± 0.038**	0.550 ± 0.062	0.606 ± 0.004	0.797 ± 0.002	0.583 ± 0.019	0.653 ± 0.003
SSL-allFeats-gr	0.530 ± 0.026	0.519 ± 0.028	**0.806 ± 0.002**	**0.974 ± 0.001**	**0.680 ± 0.020**	**0.804 ± 0.002**
SSL-nAE	0.496 ± 0.022	0.615 ± 0.023	0.530 ± 0.003	0.573 ± 0.003	0.500 ± 0.022	0.496 ± 0.003
SSL-nFeats	0.491 ± 0.028	0.510 ± 0.029	0.502 ± 0.005	0.499 ± 0.006	0.502 ± 0.037	0.499 ± 0.004
TF-C	0.511 ± 0.036	0.513 ± 0.037	0.502 ± 0.005	0.500 ± 0.006	0.509 ± 0.036	0.501 ± 0.004
TS-TCC	0.507 ± 0.025	0.499 ± 0.021	0.500 ± 0.005	0.501 ± 0.003	0.493 ± 0.028	0.500 ± 0.003
SVM*	0.559 ± 0.040	0.468 ± 0.074	0.547 ± 0.008	0.626 ± 0.042	0.567 ± 0.067	0.508 ± 0.004
ISO*	0.565 ± 0.036	0.519 ± 0.019	0.552 ± 0.006	0.496 ± 0.001	0.582 ± 0.019	0.548 ± 0.005
LOF*	0.505 ± 0.013	0.500 ± 0.010	0.502 ± 0.001	0.497 ± 0.002	0.503 ± 0.010	0.505 ± 0.001

Tabla 4.12: Valores p de cada modelo de referencia con respecto al modelo propuesto SSL, en el conjunto de datos Periódicos (ZTF-PER).

Clase Outlier	$P_{SSL-TFC}$	$P_{SSL-TCC}$	$P_{SSL-ISO}$	$P_{SSL-LOF}$	$P_{SSL-SVM}$	$P_{SSL-nAE}$	$P_{SSL-nFeats}$	$P_{SSL-allFeats}$
DSCT	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$
LPV	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$6 \cdot 10^{-1}$
RRL	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$
P-Other	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$
CEPH	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$3 \cdot 10^{-2}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$
E	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$	$1 \cdot 10^{-4}$

4.4. Comparación con Modelo MCDSVDD

En esta sección, se lleva a cabo una comparación entre el rendimiento del modelo SSL propuesto y el modelo MCDSVDD, utilizando los resultados obtenidos en el estudio previo realizado por [41]. Es relevante destacar que el modelo MCDSVDD se evalúa en el mismo conjunto de datos utilizado en este trabajo. Sin embargo, su evaluación se limita a la métrica AUROC y se basa en un conjunto de prueba con una proporción de clases *inliers* y *outliers* de 90 % y 10 %, respectivamente.

En este análisis, se considera los resultados tanto del modelo propuesto con afinamiento (SSL-FT, *SSL fine tuning*) como del modelo sin afinamiento (SSL). Se evalúa el rendimiento en dos configuraciones de balance de clases: una con una proporción equilibrada de 50 % de *inliers* y 50 % de *outliers*, y la otra con una proporción desequilibrada de 90 % de *inliers* y 10 % de *outliers*. Esta estrategia permite no solo comparar el rendimiento del modelo con el trabajo previo en [41], sino también verificar su consistencia al enfrentar escenarios más realistas, como el desbalance en la distribución de objetos astronómicos.

El proceso de afinamiento del modelo se lleva a cabo mediante la adición de una capa lineal al modelo MLP, que tiene las mismas dimensiones que la capa de salida del modelo original. Durante este proceso, se mantienen congelados los pesos del resto del modelo MLP entrenado mediante técnicas de aprendizaje auto-supervisado. De esta manera solo se realiza un afinamiento de la última capa que se ha añadido al modelo, al volver a entrenar el modelo con un subconjunto del conjunto de entrenamiento que contiene las etiquetas (30 % del conjunto de entrenamiento) y la función de costos *Crossvalidation*.

4.4.1. ZTF Transiente

En la Tabla 4.13 se muestran los resultados obtenidos por el modelo propuesto sin afinamiento (SSL), con afinamiento (SSL-FT) y MCDSVDD [41], para distintos porcentajes de desbalance de la clase *outlier* en el conjunto de prueba (50 % y 10 % de *outliers*) para el conjunto de datos transientes (ZTF-TRA). Estos resultados revelan una consistencia en el rendimiento del modelo, ya que no varía significativamente ante cambios en la distribución de datos, lo que indica su capacidad para adaptarse a diferentes proporciones de objetos astronómicos desconocidos.

Además, se observa que el afinamiento del modelo conlleva mejoras notables, especialmente para las clases SLSN y SNIbc, donde los resultados mejoran aproximadamente en un 2 %, mientras que para las demás clases, las mejoras son limitadas. Al comparar estos resultados con el modelo de referencia MCDSVDD, se destaca que el modelo propuesto supera al modelo de referencia en la mayoría de las clases de objetos transientes.

Tabla 4.13: Comparación del modelo propuesto sin afinamiento (SSL) y con afinamiento (SSL-FT), con modelo MCDSVDD para el conjunto de datos ZTF-TRA, mediante la métrica de evaluación AUROC, para distintos desbalances de clase *outlier*.

AUROC	Transiente			
Modelo	SLSN	SNII	SNIa	SNIbc
SSL 50 %	0.703 ± 0.103	0.701 ± 0.029	0.677 ± 0.019	0.617 ± 0.046
SSL-FT 50 %	0.724 ± 0.085	0.695 ± 0.026	0.669 ± 0.014	0.644 ± 0.058
SSL 10 %	0.698 ± 0.041	0.696 ± 0.029	0.644 ± 0.105	0.618 ± 0.018
SSL-FT 10 %	0.711 ± 0.027	0.683 ± 0.024	0.653 ± 0.066	0.625 ± 0.023
MCDSVDD	0.686 ± 0.051	0.828 ± 0.024	0.624 ± 0.039	0.584 ± 0.032

4.4.2. ZTF Estocástico

En la Tabla 4.14 se muestran los resultados obtenidos por el modelo propuesto sin afinamiento (SSL), con afinamiento (SSL-FT) y MCDSVDD [41], para distintos porcentajes de desbalance de la clase *outlier* en el conjunto de prueba (50 % y 10 % de *outliers*) para el conjunto de datos estocásticos (ZTF-EST). Estos resultados revelan una consistencia mayor en el rendimiento del modelo, ya que no varía significativamente ante cambios en la distribución de datos, lo que indica su capacidad para adaptarse a diferentes proporciones de objetos astronómicos desconocidos.

Además, se observa que el afinamiento del modelo conlleva mejoras notables, obteniendo mejoras de aproximadamente un 1 %, con la excepción de la clase QSO que obtiene cerca de un 6 % de mejoría al realizar el afinamiento para ambos casos de balance de datos. Al comparar estos resultados con el modelo de referencia MCDSVDD, se destaca que el modelo propuesto supera al modelo de referencia en casi todas las clases de objetos estocásticos, obteniendo resultados muy superiores en ciertas clases, como sucede para la clase CV/Nova. Sin embargo, para la clase AGN la diferencia con respecto al modelo afinado es de un 6 % por debajo al modelo de referencia.

Tabla 4.14: Comparación del modelo propuesto sin afinamiento (SSL) y con afinamiento (SSL-FT), con modelo MCDSVDD para el conjunto de datos ZTF-EST, mediante la métrica de evaluación AUROC, para distintos desbalances de clase *outlier*.

AUROC	Estocástico				
Modelo	AGN	Blazar	CV/Nova	QSO	YSO
SSL 50 %	0.628 ± 0.008	0.721 ± 0.015	0.955 ± 0.004	0.561 ± 0.021	0.867 ± 0.008
SSL-FT 50 %	0.641 ± 0.007	0.731 ± 0.016	0.969 ± 0.006	0.608 ± 0.073	0.877 ± 0.008
SSL 10 %	0.628 ± 0.005	0.720 ± 0.008	0.957 ± 0.004	0.553 ± 0.018	0.866 ± 0.004
SSL-FT 10 %	0.640 ± 0.005	0.727 ± 0.006	0.967 ± 0.003	0.611 ± 0.058	0.880 ± 0.004
MCDSVDD	0.706 ± 0.069	0.512 ± 0.113	0.770 ± 0.127	0.483 ± 0.080	0.854 ± 0.041

4.4.3. ZTF Periódico

En la Tabla 4.15 se muestran los resultados obtenidos por el modelo propuesto sin afinamiento (SSL), con afinamiento (SSL-FT) y MCDSVDD [41], para distintos porcentajes de desbalance de la clase *outlier* en el conjunto de prueba (50 % y 10 % de *outliers*) para el conjunto de datos periódicos (ZTF-PER), utilizando las 169 características asociadas a las bandas g y r (ver descripción en el Anexo B), ya que como se vio en la sección 4.3.3, los resultados para objetos periódicos mejoran al utilizar una mayor cantidad de características. Estos resultados revelan una consistencia mayor en el rendimiento del modelo, ya que no varía significativamente ante cambios en la distribución de datos, lo que indica su capacidad para adaptarse a diferentes proporciones de objetos astronómicos desconocidos.

Además, al igual que para los casos anteriores, se observa que el afinamiento del modelo conlleva mejoras notables, con la excepción de la CEPH. Si bien, los resultados obtenidos con afinamiento para determinadas clases son buenos, como es el caso de las clases RRL, LPV y E, al comparar estos resultados con el modelo de referencia MCDSVDD, se observa que el modelo de referencia supera al modelo propuesto en la todas las clases de objetos periódicos con excepción de la clase LPV, donde la diferencia es de alrededor de un 2 %.

A pesar de que los resultados mejoraron bastante al incluir características más relevantes obtenidas al combinar las bandas de observación g y r, como se indica en [42], los resultados pueden no ser tan buenos en comparación con el modelo de referencia MCDSVDD, debido a pérdidas de información de las curvas de luz a la hora de codificar estas ultimas a un vector de largo fijo. Esto sumado al hecho que el desbalance de clases en el conjunto de entrenamiento es bastante grande, por lo que el autoencoder no es capaz de aprender a codificar el período

de todas las clases periódicas minoritarias.

Tabla 4.15: Comparación del modelo propuesto sin afinamiento (SSL) y con afinamiento (SSL-FT), con modelo MCDSVDD para el conjunto de datos ZTF-PER, mediante la métrica de evaluación AUROC, para distintos desbalances de clase *outlier*.

AUROC	Periódico				
Modelo	DSCT	RRL	LPV	CEPH	E
SSL 50 %	0.530 ± 0.026	0.806 ± 0.002	0.974 ± 0.001	0.680 ± 0.020	0.804 ± 0.002
SSL-FT 50 %	0.504 ± 0.017	0.854 ± 0.005	0.977 ± 0.000	0.654 ± 0.019	0.839 ± 0.003
SSL 10 %	0.538 ± 0.015	0.805 ± 0.006	0.974 ± 0.002	0.670 ± 0.011	0.802 ± 0.002
SSL-FT 10 %	0.502 ± 0.008	0.857 ± 0.005	0.977 ± 0.002	0.662 ± 0.014	0.840 ± 0.004
MCDSVDD	0.819 ± 0.015	0.953 ± 0.003	0.953 ± 0.008	0.858 ± 0.025	0.945 ± 0.006

Capítulo 5

Conclusión

En la sección 4.1.1, el entrenamiento del AE se realizó utilizando todos los conjuntos de datos a la vez, sin separar por clases transientes, estocásticas y periodicas, para posteriormente realizar un afinamiento del modelo con solo los datos pertenecientes al respectivo conjunto de datos. Esto debido a la variedad de puntos con los que cuenta cada curva de luz, que van de curvas con un mínimo de 5 observaciones temporales, hasta más de 100 puntos, lo cual dificulta el aprendizaje del modelo de aprendizaje auto-supervisado. Por lo que el uso del AE, como se observó en la sección 4.1.2, mejora el rendimiento del modelo SSL considerablemente. Esto nos dice que a pesar del sesgo intrínseco generado por el desbalance de clases y la variabilidad en los muestreos de las curvas de luz, el modelo AE fue capaz de encontrar buenas representaciones de la información proveniente de las curvas originales, de un largo fijo, pero a costa de pérdida de información como es el caso de curvas de luz periódicas.

Para abordar esta situación, se consideró la inclusión de características derivadas de ambas bandas de observación g y r, siguiendo el enfoque presentado en [42]. Este estudio sugiere que una de las características más efectivas para la clasificación de objetos periódicos es el período multibanda, el cual se obtiene al combinar información de las bandas de observación g y r, y como se observó en la sección 4.3.3, los resultados obtenidos por el modelo de aprendizaje auto-supervisado aumentan significativamente. Tambíen, dentro del trabajo presentado en [42] se indica que para curvas de luz estocásticas, las características que aportan más para la identificación de clases corresponden a atributos de color, de los cuales algunos de ellos provienen del catálogo AllWISE [49] [50], por lo que la inclusión de estas características podría mejorar el rendimiento del modelo.

Como se observó en el capitulo 4, el modelo propuesto fue superior a los modelos basados en características, para la mayoría de las clases de los conjuntos de datos, especialmente para clases que cuentan con una menor cantidad de datos. Si bien, los resultados obtenidos para clases con mayor población o frecuencia dentro del conjunto de datos, no fueron muy superiores a los obtenidos por los modelos base, el objetivo general de la investigación era desarrollar un modelo de *deep learning* basado en aprendizaje auto-supervisado para la detección de curvas de luz atípicas o novedosas, por lo que se puede concluir que el modelo implementado cumple con este objetivo y demuestra la efectividad de este tipo de modelos para identificar objetos astronómicos anómalos a partir de su curva de luz.

En cuanto al uso del test estadístico, se puede decir que este enfoque se mostró útil para evaluar si el modelo SSL superaba de manera significativa a los modelos de referencia, y si dicha diferencia estaba respaldada de manera estadística por los datos, ya que si la diferencia observada en los puntajes resultaba sustancialmente mayor de lo que podría esperarse por azar, entonces se podía rechazar la hipótesis nula (H_0) y concluir que el modelo SSL exhibía un rendimiento significativamente superior.

Dentro de las ventajas de los modelos SSL por sobre otros tipos de aprendizajes, es que estos no requieren del uso de datos etiquetados o previamente clasificados para su entrenamiento, por lo que se podría utilizar la gran cantidad de datos de objetos astronómicos que se recopilan a diario, que no se encuentran clasificados, para el entrenamiento del modelo en conjunto con los que se encuentran etiquetados, para posteriormente realizar un afinamiento de los datos con un subconjunto de datos etiquetados con el objetivo de mejorar los resultados obtenidos.

Es importante mencionar que este estudio tiene ciertas limitaciones que deben tenerse en cuenta al interpretar los resultados. Por ejemplo, las conclusiones se basan en conjuntos de datos específicos utilizados en el estudio, lo que podría limitar la generalización de los resultados a otros conjuntos de datos astronómicos. Además, se puede dar el caso de que el objeto desconocido que se quiere identificar mediante el modelo, puede tener atributos similares a otras clases que fueron utilizadas para el entrenamiento del modelo SSL, por lo que este podría asignar un puntaje bajo de anomalía.

La hipótesis planteada de implementar un modelo basado en aprendizaje auto-supervisado capaz de identificar curvas de luz anómalas ha sido corroborada exitosamente. Los resultados obtenidos demuestran que este tipo de modelos puede aprender características o patrones distintivos a partir de series temporales de longitud variable y muestreadas de forma irregular, incluso sin utilizar etiquetas o clases predefinidas. Esto valida la capacidad de los modelos de aprendizaje auto-supervisado para abordar la tarea compleja de detectar curvas de luz anómalas en el campo de la astronomía.

Además, se ha demostrado que estos modelos pueden asignar puntajes de anomalía a objetos astronómicos, de manera efectiva principalmente en clases con menor cantidad de datos. Esta capacidad de asignar puntajes de anomalía es valiosa en el contexto de la astronomía, donde la detección de eventos astronómicos raros o inusuales es fundamental para la investigación y comprensión del universo.

5.1. Trabajo Futuro

El método y modelo propuesto presentan oportunidades de mejora en varias etapas. Una mejora potencial para el modelo es la inclusión del vector de características como un cuarto canal desde el inicio, ya que permitiría que el modelo aprende desde el principio a conservar solo las características que son realmente útiles para el agrupamiento de datos en sus respectivas clases. Al incorporar el vector de características como una entrada adicional, se podría mejorar la capacidad del modelo para capturar características relevantes y distinguir entre objetos conocidos y anomalías.

Otra área que podría beneficiarse de mejoras es la inclusión de características de colores de AllWISE [49] [50]. En el presente trabajo, al incorporar características multibanda, se utilizan atributos de color de los objetos astronómicos, pero no se abarcan todos los que podrían obtenerse del catálogo de AllWISE. Como se demostró en [42], las características multibanda y de colores desempeñan un papel significativo en la clasificación de clases, lo que podría potencialmente mejorar en gran medida el rendimiento del modelo para la identificación de clases desconocidas.

Como se discutió en el capítulo anterior, el modelo basado en aprendizaje auto-supervisado puede tener dificultades cuando se entrena con clases que presentan un gran desbalance, principalmente por el enfoque contrastivo que tiene el entrenamiento del modelo. Esto se debe a que el modelo puede tener dificultades para aprender características distintivas que permitan formar *clusters* y discriminar entre objetos conocidos y anomalías. Para abordar este desafío, se sugiere explorar el efecto del entrenamiento del modelo con clases balanceadas utilizando técnicas como *oversampling*, *subsampling* o la generación de curvas de luz sintéticas como en el trabajo realizado en [51], debido a que el uso de *data augmentation* para introducir variabilidad durante el entrenamiento del modelo, no fue suficiente para compensar el desbalance entre las clases de cada conjunto de datos utilizado.

El uso de técnicas de *oversampling* implicaría aumentar la cantidad de muestras de las clases minoritarias mediante la replicación o generación de instancias sintéticas. Esto permitiría equilibrar la distribución de clases y proporcionar al modelo una representación más equitativa de las clases, lo que podría mejorar su capacidad para detectar y distinguir objetos anómalos.

Por otro lado, el *subsampling* implicaría reducir la cantidad de muestras de las clases mayoritarias para equilibrar la distribución de clases. Esto permitiría al modelo enfocarse más en las clases minoritarias y aprender características distintivas específicas de ellas.

La generación de curvas de luz sintéticas es otra estrategia que se puede explorar para abordar el desequilibrio de clases. Esta técnica implicaría generar datos sintéticos que se asemejen a las curvas de luz de las clases minoritarias. Esto permitiría aumentar la cantidad de muestras para las clases minoritarias y, nuevamente, proporcionar una representación más equilibrada de las clases al modelo, como en [51].

Otro aspecto importante a mejorar es el preprocesamiento de los datos. Es necesario considerar métodos alternativos de procesamiento que permitan mitigar el impacto de la cantidad de puntos que conforman las curvas de luz. Este aspecto es especialmente relevante debido a que los conjuntos de datos utilizados presentan una gran variabilidad en el número de puntos u observaciones para cada objeto astronómico.

La disparidad en la cantidad de puntos de las curvas de luz puede afectar significativamente el entrenamiento del modelo y su capacidad para encontrar características o patrones distintivos de cada clase en los conjuntos de datos. Por ejemplo, cuando se comparan curvas con solo 5 puntos con curvas que contienen 100 o más puntos, el modelo puede tener dificultades para capturar la variabilidad y los detalles finos de las curvas con menos puntos. Esto puede resultar en una pérdida de información importante y afectar la capacidad del modelo para discriminar entre objetos conocidos y anomalías.

Bibliografía

[1] S. Haykin, *Neural Networks and Learning Machines*. Pearson, Prentice Hall, 2009.

[2] X. Zhang, Z. Zhao, T. Tsiligkaridis, and M. Zitnik, "Self-supervised contrastive pre-training for time series via time-frequency consistency," in *Proceedings of Neural Information Processing Systems, NeurIPS*, 2022.

[3] E. Eldele, M. Ragab, Z. Chen, M. Wu, *et al.*, "Self-supervised contrastive representation learning for semi-supervised time-series classification," *arXiv:2208.06616*, 2022.

[4] M. Romero, "Detección de novedades en curvas de luz basado en aprendizaje de máquinas," *Tesis de Magíster en Ciencias de la Ingeniería. mención Eléctrica, Universidad de Chile*, 2022.

[5] X. Yang, Z. Zhang, and R. Cui, "Timeclr: A self-supervised contrastive learning framework for univariate time series representation," *Knowledge-Based Systems*, vol. 245, p. 108606, 2022.

[6] A. van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *arXiv: 1807.03748*, 2019.

[7] N. Vijayrania, "Self-supervised learning methods for computer vision," *Medium*, Dec 2020. Available: `https://towardsdatascience.com/self-supervised-learning-methods-for-computer-vision-c25ec10a91bd`.

[8] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," *arXiv:1911.05722*, 2020.

[9] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," *arXiv:2002.05709*, 2020.

[10] M. Masana, I. Ruiz, J. Serrat, J. van de Weijer, and A. M. López, "Metric learning for novelty and anomaly detection," *CoRR*, vol. abs/1808.05492, 2018.

[11] "Light curves and what they can tell us," *NASA*, 2013. Available: `https://imagine.gsfc.nasa.gov/science/toolbox/timing1.html`.

[12] F. R. Chromey, *To measure the sky: An introduction to observational astronomy*. Cambridge University Press, 2010.

[13] J. R. Percy, *Understanding variable stars*. Cambridge University Press, 2011.

[14] "La espectroscopia en la astronomía," *AstroMía*. Available: https://www.astromia.com/historia/espectrohistoria.html.

[15] "What is supervised learning?," *IBM Documentation*, 2023. Available: https://www.ibm.com/topics/supervised-learning#:~:text=Supervised%20learning%20uses%20a%20training,error%20has%20been%20sufficiently%20minimized.

[16] S. Bansal, "Supervised and unsupervised learning," *GeeksforGeeks*, Apr 2023. Available: https://www.geeksforgeeks.org/supervised-unsupervised-learning/.

[17] J. Brownlee, "14 different types of learning in machine learning," *Machine Learning Mastery*, Nov 2019. Available: https://machinelearningmastery.com/types-of-learning-in-machine-learning/.

[18] F. Hvilshøj, "Self-supervised learning explained," *Encord*, 2023. Available: https://encord.com/blog/self-supervised-learning/#h1.

[19] D. Shah, "Self-supervised learning and its applications," *Neptune.ai*, 2023. Available: https://neptune.ai/blog/self-supervised-learning.

[20] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, (Minneapolis, Minnesota), pp. 4171–4186, Association for Computational Linguistics, June 2019.

[21] G. Boesch, "Self-supervised learning: Everything you need to know," *Viso.ai*, 2023. Available: https://viso.ai/deep-learning/self-supervised-learning-for-computer-vision/.

[22] IBM, "What is a neural network?," *IBM Documentation*. Available: https://www.ibm.com/topics/neural-networks.

[23] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain.," *Psychological Review*, vol. 65, no. 6, p. 386–408, 1958.

[24] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature News*. Available: https://www.nature.com/articles/323533a0.

[25] S. Weidman, *Deep learning from scratch: Building with python from first principles*. O'Reilly Media, Inc, 2019.

[26] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[27] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, p. 1735–1780, 1997.

[28] S. Dobilas, "Lstm recurrent neural networks-how to teach a network to remember the past," *Medium*, Mar 2022. Available: urlhttps://towardsdatascience.com/lstm-recurrent-neural-networks-how-to-teach-a-network-to-remember-the-past-55e54c2ff22e.

[29] "K-means, clustering," *Scikit-Learn*. Available: https://scikit-learn.org/stable/modules/clustering.html#k-means.

[30] F. Sanz, "Algoritmo k-means clustering – aplicaciones y desventajas," *The Machine Learners*. Available: https://www.themachinelearners.com/k-means/.

[31] "K-nearest neighbors algorithm," *IBM Documentation*. Available: https://www.ibm.com/topics/knn#:~:text=The%20k%2Dnearest%20neighbors%20algorithm%2C%20also%20known%20as%20KNN%20or,of%20an%20individual%20data%20point.

[32] "Resumir: Estadísticos," *IBM Documentation*, 2023. Available: https://www.ibm.com/docs/es/spss-statistics/saas?topic=summarize-statistics.

[33] C. C. Aggarwal, *Outlier Analysis*. Springer Publishing Company, Incorporated, 2nd ed., 2016.

[34] J. Han, M. Kamber, and J. Pei, "12 - outlier detection," in *Data Mining (Third Edition)* (J. Han, M. Kamber, and J. Pei, eds.), The Morgan Kaufmann Series in Data Management Systems, pp. 543–584, Boston: Morgan Kaufmann, third edition ed., 2012.

[35] H. Jagadish, N. Koudas, and S. Muthukrishnan *Mining Deviants in a Time-Series Database*, 1999.

[36] D. J. Cook and L. B. Holder, "Graph-based data mining," *IEEE Intelligent Systems*, vol. 15, p. 32–41, mar 2000.

[37] V. Kilaru, "One class classification using support vector machines," *Analytics Vidhya*, Jun 2022. Available: https://www.analyticsvidhya.com/blog/2022/06/one-class-classification-using-support-vector-machines/.

[38] A. Fernández Jauregui, "Isolation forest en python," *anderfernandez*, Feb 2023. Available: https://anderfernandez.com/blog/isolation-forest-en-python/.

[39] "Outlier detection with local outlier factor (lof)," *Scikit Learn*, 2011. Available: https://scikit-learn.org/stable/auto_examples/neighbors/plot_lof_outlier_detection.html.

[40] V. Jayaswal, "Local outlier factor (lof)-algorithm for outlier identification," *Medium*, Nov 2020. Available: https://towardsdatascience.com/local-outlier-factor-lof-algorithm-for-outlier-identification-8efb887d9843.

[41] M. Pérez-Carrasco, G. Cabrera-Vives, L. Hernández-García, F. Forster, *et al.*, "Alert classification for the alerce broker system: The anomaly detector," 2023.

[42] P. Sánchez-Sáez, I. Reyes, C. Valenzuela, F. Förster, *et al.*, "Alert classification for the alerce broker system: The light curve classifier," *The Astronomical Journal*, vol. 161, p. 141, feb 2021.

[43] G. W. Cobb, *Introduction to design and analysis of experiments*. Wiley, 2015.

[44] G. Iglesias, E. Talavera, Á. González-Prieto, A. Mozo, *et al.*, "Data augmentation techniques in time series domain: a survey and taxonomy," *Neural Computing and Applications*, vol. 35, pp. 10123–10145, May 2023.

[45] F. Förster, G. Cabrera-Vives, E. Castillo-Navarrete, P. A. Estévez, *et al.*, "The automatic learning for the rapid classification of events (alerce) alert broker," *The Astronomical Journal*, vol. 161, p. 242, apr 2021.

[46] R. Carrasco-Davis, E. Reyes, C. Valenzuela, F. Förster, *et al.*, "Alert classification for the alerce broker system: The real-time stamp classifier," *The Astronomical Journal*, vol. 162, p. 231, nov 2021.

[47] N. Prantzos, *Supernova Types*, pp. 2442–2442. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015.

[48] "Features," *ALeRCE Science*. Available: `https://alerce.science/features/`.

[49] E. L. Wright, P. R. M. Eisenhardt, A. K. Mainzer, M. E. Ressler, *et al.*, "The wide-field infrared survey explorer (wise): Mission description and initial on-orbit performance," *The Astronomical Journal*, vol. 140, p. 1868, nov 2010.

[50] A. Mainzer, J. Bauer, T. Grav, J. Masiero, *et al.*, "Preliminary results from neowise: An enhancement to the wide-field infrared survey explorer for solar system science," *The Astrophysical Journal*, vol. 731, p. 53, mar 2011.

[51] V. Caragol, "Optimización de técnicas de balance de datos para clasificador de curvas de luz basado en xgboost," *Memoria de Ingeniería Civil Eléctrica, Universidad de Chile*, 2023.

[52] J. Davis and M. Goadrich, "The relationship between precision-recall and roc curves," in *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, (New York, NY, USA), p. 233–240, Association for Computing Machinery, 2006.

[53] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

Anexos

A. Acrónimos

Listado alfabético de los acrónimos utilizados en esta tesis:

- A: área a través de la cual se recibe la energía.
- AE: modelo autoencoder.
- AGN: núcleo galáctico activo.
- ANN: redes neuronales artificiales, del inglés artificial neural networks.
- AUCPR: área bajo la curva de precisión y exhaustividad.
- AUROC: área bajo la curva de característica operativa del receptor.
- BERT: modelo Bidirectional Encoder Representations from Transformers.
- BLA: blazar.
- CEPH: estrella pulsante periódica Cefeida.
- CNN: red neuronal convolucional, del inglés Convolutional Neural Network.
- CV: estrella variable cataclísmica.
- DSCT: estrella pulsante delta scuti o cefeida enana.
- E: energía total recibida desde la fuente de origen.
- E/EW: estrella binaria eclipsante.
- EM: maximización de la esperanza.
- EVA: análisis de valores extremos.
- F: brillo aparente o flujo.
- FPR: tasa de falsos positivos (false positive rate).
- H_0: hipótesis nula.
- H_1: hipótesis alternativa.

- ISO: modelo isolation forest.
- K: medida de curtosis, del inglés Kurtosis.
- K-Means: método de cuantificación de vectores K-Medias (K-Means).
- KNN: K-Vecinos más cercanos (K-Nearest Neighbors).
- LC: curva de luz (light curve).
- LOF: modelo local outlier Factor.
- LPV: estrella variable de periodo largo.
- LRD: desviación de densidad local.
- LSTM: red de memoria a corto-largo plazo (long-short term memory).
- m: magnitud aparente.
- MCDSVDD: modelo Multi-Class Deep Support Vector Data Description.
- MJD: fecha juliana modificada.
- ML: aprendizaje de máquinas, del inglés Machine Learning.
- MLP: modelo perceptrón multicapa (Multilayer Perceptron).
- MSE: error cuadrático medio (Mean Square Error).
- P-Other: otros tipos de estrellas variables.
- PCA: análisis de componentes esenciales.
- QSO: cuásar o quasar.
- RD: alcance (Reachability Distance).
- RNN: red neuronal recurrente, del ingles Recurrente Neural Network.
- ROC: característica operativa del receptor (Receiver Operating Characteristic).
- RRL: estrella pulsante RR Lyrae.
- SI: sistema internacional de unidades.
- SimCLR: modelo Simple Framework for Contrastive Unsupervised Representation Learning.
- SLSN: supernova súper lumínica.
- SNIa: supernova tipo Ia.
- SNIbc: supernova tipo Ib o Ic.
- SNII: supernova tipo II.
- SSL: aprendizaje auto-supervisado, del inglés Self Supervised Learning.
- SSL-FT: modelo propuesto de aprendizaje auto-supervisado, con afinamiento en la ultima capa lineal.

- SSL-nAE: modelo propuesto de aprendizaje auto-supervisado, pero que no hace uso de autoencoder.
- SSL-nFeats: modelo propuesto de aprendizaje auto-supervisado, pero que no hace uso de vector de características.
- SVM: support vector Machine.
- t: tiempo durante el cual se recibe la energía.
- TF-C: modelo de consistencia tiempo-frecuencia (Time-Frequency Consistency).
- TPR: tasa de verdaderos positivos (true positive rate).
- TS-TCC: modelo Time-Series representation learning framework via Temporal and Contextual Contrasting.
- YSO: objeto estelar joven.
- ZTF: Conjunto de datos de objetos astronómicos Zwicky Transient Facility.
- ZTF-EST: Conjunto de datos con curvas de luz estocásticas provenientes del dataset ZTF.
- ZTF-PER: Conjunto de datos con curvas de luz periódicas provenientes del dataset ZTF.
- ZTF-TRA: Conjunto de datos con curvas de luz transientes provenientes del dataset ZTF.

B. Descripción de Características

B.1. Características de Banda Única

A continuación, se detalla cada una de las características computadas a partir de las bandas de observación g y r. Cabe resaltar que estos cálculos se efectúan de manera independiente para cada banda:

- MHPS_ratio: Razón entre las varianzas MHPS_low y MHPS_high para una banda dada.
- MHPS_low: Varianza asociada a una escala de tiempo de 100 días obtenida de un análisis MHPS.
- MHPShigh: Varianza asociada a una escala de tiempo de 10 días obtenida de un análisis MHPS.
- MHPS_non_zero: Número de puntos en la curva de luz utilizados para el análisis MHPS.
- MHPS_PN_flag: Indicador que informa si el ruido de Poisson es mayor que la varianza MHPS_high.

- iqr: Rango intercuartílico, que es la diferencia entre el tercer y el primer cuartil de la curva de luz.
- Amplitude: Mitad de la diferencia entre la mediana de las magnitudes máximas y mínimas del 5 % superior e inferior de las magnitudes.
- AndersonDarling: Prueba para verificar si una muestra de datos proviene de una población con una distribución específica, en este caso, una distribución normal.
- Autocor_length: Valor de rezago donde la función de autocorrelación se vuelve menor que Eta_e.
- Beyond1Std: Porcentaje de puntos con magnitudes fotométricas que se encuentran más allá de 1 desviación estándar de la media.
- Con: Número de tres puntos de datos consecutivos más brillantes o más tenues que 2 desviaciones estándar de la curva de luz.
- Eta_e: Relación entre la media de los cuadrados de las diferencias de magnitud sucesivas y la varianza de la curva de luz.
- Gskew: Medida basada en la mediana que evalúa la asimetría de la curva de luz.
- MaxSlope: Máxima pendiente absoluta entre dos observaciones consecutivas de magnitud.
- Mean: Magnitud media de la curva de luz (o magnitud media de la diferencia si la fuente no puede corregirse).
- Meanvariance: Razón entre la desviación estándar y la magnitud media de la curva de luz.
- MedianAbsDev: Discrepancia mediana entre los datos y la mediana de los datos.
- MedianBRP: Fracción de puntos fotométricos dentro de la amplitud/10 de la magnitud mediana.
- PairSlopeTrend: Fracción de primeras diferencias crecientes menos la fracción de primeras diferencias decrecientes en las últimas 30 medidas de magnitud ordenadas en el tiempo.
- PercentAmplitude: Mayor diferencia porcentual entre la magnitud máxima o mínima y la magnitud mediana.
- Q31: Diferencia entre el tercer y el primer cuartil de la curva de luz.
- Rcs: Rango de una suma acumulativa.
- Skew: Medida de asimetría de la curva de luz.
- SmallKurtosis: Curtosis de la muestra de magnitudes calculada para muestras pequeñas.
- Std: Desviación estándar de la curva de luz.
- StetsonK: Medida robusta de la curtosis.
- Pvar: Probabilidad de que la fuente sea intrínsecamente variable.

- ExcessVar: Medida de la amplitud de la variabilidad intrínseca.
- SF_ML_amplitude: Diferencia de magnitud RMS de la función de estructura calculada en una escala de tiempo de 1 año.
- SF_ML_gamma: Gradiente logarítmico del cambio medio en magnitud calculado a partir de la función de estructura.
- IAR_phi: Nivel de autocorrelación utilizando una representación discreta de tiempo de un modelo DRW.
- LinearTrend: Pendiente de un ajuste lineal a la curva de luz.
- delta_mag_fid: Diferencia entre la magnitud máxima y mínima observada en una banda dada.
- delta_mjd_fid: Duración total de la curva de luz en una banda dada.
- first_mag: Magnitud de la primera alerta en una banda dada.
- mean_mag: Magnitud media de la curva de luz de la alerta en una banda dada.
- min_mag: Magnitud mínima de la curva de luz de la alerta en una banda dada.
- n_det: Número de detecciones en la curva de luz de la alerta en una banda dada.
- n_neg: Número de detecciones negativas en la curva de luz de la alerta (isdiffpos =- 1).
- n_pos: Número de detecciones positivas en la curva de luz de la alerta (isdiffpos =+ 1).
- positive_fraction: Fracción de detecciones en las imágenes de diferencia de una banda dada que son más brillantes que la imagen de plantilla.
- n_non_det_before_fid: Número de no detecciones en la banda 'x' antes de la primera detección en cualquier banda.
- max_diffmaglim_before_fid: Límite de diferencia máximo de no detección en la banda "x" antes de la primera detección en cualquier banda.
- median_diffmaglim_before_fid: Límite de diferencia mediano de no detección en la banda '"x" antes de la primera detección en cualquier banda.
- last_diffmaglim_before_fid: Último límite de diferencia de no detección en la banda "x" antes de la primera detección en cualquier banda.
- last_mjd_before_fid: Última Fecha Juliana Modificada (MJD) de no detección en la banda "x" antes de la primera detección en cualquier banda.
- dmag_non_det_fid: Diferencia entre el límite de diferencia de no detección mediano en la banda "x" antes de la primera detección y la magnitud mínima detectada (pico) en la banda "x".
- dmag_first_det_fid: Diferencia entre el último límite de diferencia de no detección en la banda "x" antes de la primera detección en cualquier banda y la primera magnitud detectada en la banda "x".

- n_non_det_after_fid: Número de no detecciones en la banda "x" después de la primera detección en cualquier banda.
- max_diffmaglim_after_fid: Límite de diferencia máximo de no detección en la banda "x" después de la primera detección en cualquier banda.
- median_diffmaglim_after_fid: Límite de diferencia mediano de no detección en la banda "x" después de la primera detección en cualquier banda.
- SPM_A: Modelo paramétrico de supernova A.
- SPM_t0: Modelo paramétrico de supernova t_0.
- SPM_gamma: Modelo paramétrico de supernova gamma.
- SPM_beta: Modelo paramétrico de supernova beta.
- SPM_tau_rise: Tiempo de subida del modelo paramétrico de supernova.
- SPM_tau_fall: Tiempo de caída del modelo paramétrico de supernova.
- SPM_chi: Chi cuadrado reducido del ajuste de la curva de luz del modelo paramétrico de supernova.
- Period_band: Período de una sola banda calculado utilizando un periodograma de Análisis Armónico Múltiple de Varianza (MHAOV).
- delta_period: Valor absoluto de la diferencia entre el período Multiband y el período MHAOV obtenido utilizando una sola banda.
- Psi_CS: Rango de una suma acumulativa aplicada a la curva de luz plegada en fase.
- Psi_eta: Índice Eta_e calculado a partir de la curva de luz plegada.
- Harmonics_mag_1 a Harmonics_mag_7: Amplitud de los componentes de la serie armónica (obtenida mediante el ajuste de una serie armónica hasta el séptimo armónico).
- Harmonics_phase_2 a Harmonics_phase_7: Fase de los componentes de la serie armónica (obtenida mediante el ajuste de una serie armónica hasta el séptimo armónico).
- Harmonics_mse: Error cuadrático medio de la serie armónica (obtenido mediante el ajuste de una serie armónica hasta el séptimo armónico).
- GP_DRW_sigma: Amplitud de la variabilidad a escalas de tiempo cortas (t $\ll$ tau), obtenida del modelo DRW (Damped Random Walk).
- GP_DRW_tau: Tiempo de relajación (tau) obtenido del modelo DRW (Damped Random Walk).

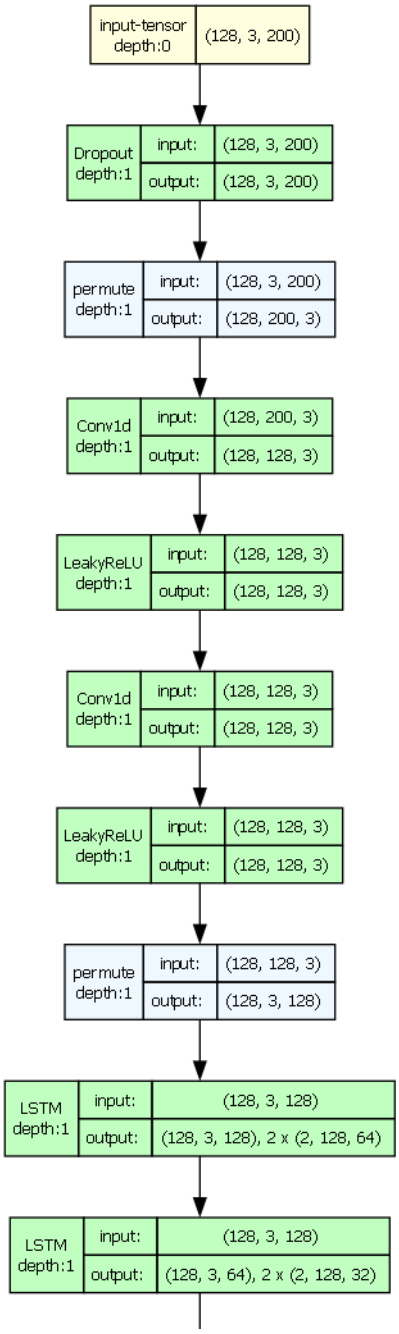
B.2. Características Multibanda

Las características presentadas a continuación han sido calculadas mediante un análisis conjunto de las bandas de observación g y r:

- g-r_max: Color g-r obtenido utilizando la magnitud más brillante de la curva de luz diferencial (lc_diff) en cada banda.
- g-r_mean: Color g-r obtenido utilizando la magnitud media de la curva de luz diferencial en cada banda.
- g-r_max_corr: Color g-r obtenido utilizando la magnitud más brillante de la curva de luz corregida (lc_corr) en cada banda.
- g-r_mean_corr: Color g-r obtenido utilizando la magnitud media de la curva de luz corregida en cada banda.
- Power_rate: Proporción entre la potencia del periodograma multibanda obtenido para el mejor candidato a periodo (P).
- Power_rate_2: Proporción entre la potencia del periodograma multibanda obtenido para el mejor candidato a periodo (P) y para $2P$.
- Power_rate_3: Proporción entre la potencia del periodograma multibanda obtenido para el mejor candidato a periodo (P) y para $3P$.
- Power_rate_4: Proporción entre la potencia del periodograma multibanda obtenido para el mejor candidato a periodo (P) y para $4P$.
- Power_rate_1/2: Proporción entre la potencia del periodograma multibanda obtenido para el mejor candidato a periodo (P) y para $\frac{P}{2}$.
- Power_rate_1/3: Proporción entre la potencia del periodograma multibanda obtenido para el mejor candidato a periodo (P) y para $\frac{P}{3}$.
- Power_rate_1/4: Proporción entre la potencia del periodograma multibanda obtenido para el mejor candidato a periodo (P) y para $\frac{P}{4}$.
- Multiband_period: Período obtenido utilizando el periodograma MHAOV multibanda.
- PPE: Pseudo Entropía del Periodograma Multibanda.

C. Arquitectura Modelo Propuesto

C.1. Arquitectura Modelo Autoencoder



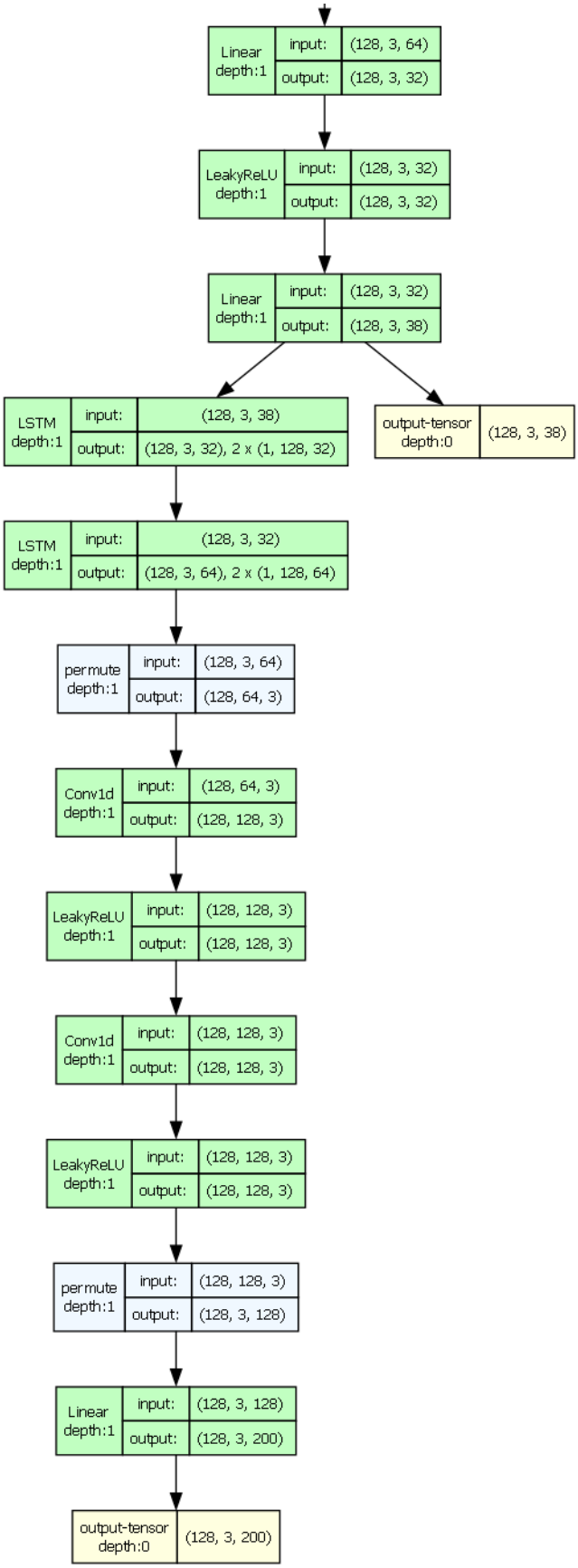


Figura 5.1: Arquitectura detallada del modelo Autoencoder.

C.2. Arquitectura Modelo MLP Aprendizaje Auto-Supervisado

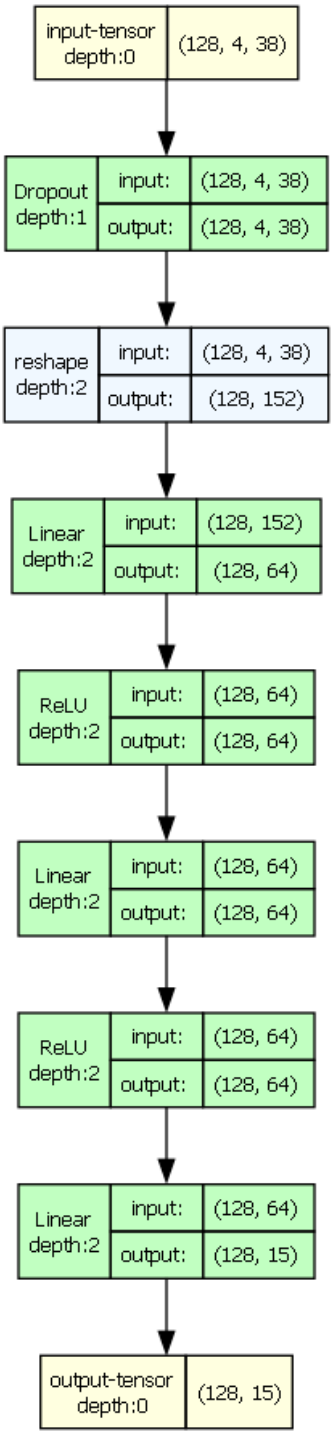


Figura 5.2: Arquitectura detallada del modelo MLP entrenado mediante aprendizaje auto-supervisado.