UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

# APRENDIZAJE PROFUNDO PARA LA CLASIFICACIÓN Y DETECCIÓN DE ANOMALÍAS EN IMÁGENES DE OBJETOS ASTRONÓMICOS

TESIS PARA OPTAR AL GRADO DE
MAGÍSTER EN CIENCIAS DE LA INGENIERÍA, MENCIÓN ELÉCTRICA

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL ELÉCTRICO

ESTEBAN DIRK REYES DE JONG

PROFESOR GUÍA:
PABLO ANTONIO ESTÉVEZ VALENCIA

MIEMBROS DE LA COMISIÓN:
PABLO ZEGERS FERNÁNDEZ
CLAUDIO PEREZ FLORES

SANTIAGO DE CHILE
2021

RESUMEN DE LA TESIS PARA OPTAR AL
GRADO DE MAGÍSTER EN CIENCIAS DE LA
INGENIERÍA, MENCIÓN ELÉCTRICA
Y AL TÍTULO DE INGENIERO CIVIL ELÉCTRICO
POR: ESTEBAN DIRK REYES DE JONG
FECHA: 2021
PROF. GUÍA: PABLO ANTONIO ESTÉVEZ VALENCIA

# APRENDIZAJE PROFUNDO PARA LA CLASIFICACIÓN Y DETECCIÓN DE ANOMALÍAS EN IMÁGENES DE OBJETOS ASTRONÓMICOS

En esta tesis se trabaja sobre tres temas principales. El primero es la clasificación de objetos astronómicos utilizando alertas generadas por el Zwicky Transient Facility (ZTF). Se propone la utilización de imágenes y metadatos contenidos en la primera detección de objetos astronómicos para entrenar una red neuronal convolucional. El clasificador denominado stamp classifier es capaz de clasificar entre 5 clases, logrando un accuracy de 94.1 %. Segundo, se propone una serie de mejoras a un modelo de detección de anomalías llamado GeoTransform, basado en aprendizaje profundo y transformaciones geométricas sobre imágenes, para mejorar su desempeño en la detección de bogus (artefactos u objetos que no son de interés astrofísico). Se proponen nuevas transformaciones y estrategias para la selección de transformaciones, las que son probadas sobre las bases de datos High Cadence Transient Survey (HiTS) y ZTF, obteniéndose un área bajo la curva ROC (Receiver Operating Characteristic) de 99.42 % para HiTS y 94.14 % para ZTF. En la tercera parte se miden los efectos de utilizar bogus etiquetados de forma automática por GeoTransform en vez de bogus etiquetados manualmente para el entrenamiento del stamp classifier sobre la base ZTF, obteniéndose un aumento del 7.74 %, en el accuracy del stamp classifier.

*A mi madre y mi padre. Muchas gracias.*

# Agradecimientos

Este trabajo fue parcialmente financiado por la Agencia Nacional de Investigación y Desarrollo (ANID) a través de la beca MAGÍSTER NACIONAL/2019 - 22190947 y el proyecto FONDECYT - 1171678. También se agradece el financiamiento de la Iniciativa Científica Instituto Milenio del Ministerio Chileno de Economía, Desarrollo y Turismo, a través de la beca IC12009 otorgada al Instituto Milenio de Astrofísica (MAS, por sus siglas en inglés *Millennium Institute of Astrophysics*). Adicionalmente, el autor agradece el financiamiento para asistir a conferencia internacionales otorgado por el Departamento de Ingeniería Eléctrica de la Universidad de Chile.

# Tabla de Contenido

# Índice de Tablas

# Índice de Ilustraciones

# Capítulo 1

# Introducción

La astronomía ha recientemente entrado en la era de los datos masivos (*big data*), debido a la construcción de enormes observatorios, capaces de producir imágenes astronómicas de regiones cada vez más amplias del cielo y a alta cadencia (alta frecuencia de muestreo). Un ejemplo de este tipo de instalaciones son el *Vera C. Rubin Observatory* (Rubin Obs [6]) y el *Zwicky Transient Facility* (ZTF [7]), estos telescopios tienen como uno de sus principales objetivos el estudio de objetos astronómicos en el dominio del tiempo, como lo son las supernovas, fenómeno transitorio cuyo análisis resulta útil para la generación de modelos astrofísicos o cosmología [8, 9]. La alta cadencia y la profundidad de observación de estos estudios astronómicos (*surveys*) permitirán encontrar objetos nunca antes detectados por otros telescopios, como lo son los fenómenos de corta duración y particularmente lejanos. Idealmente, se busca que los objetos transitorios sean detectados lo antes posible, con el objetivo de medir su información en frecuencia con mayor resolución, a través de telescopios de seguimiento (*follow-up*). Mientras antes se realiza este seguimiento, mejor se caracteriza el fenómeno ocurrido. Debido a la necesidad de procesar estos datos, actualmente existen proyectos en desarrollo llamados *astronomical alert brokers* (e.g. ALeRCE [3], ANTARES [10], Lasair [11]), encargados de recibir, procesar, clasificar y reportar los objetos en el flujo (*stream*) de datos entregados por los observatorios, con el fin de que sean de fácil acceso para la comunidad astronómica. La presente tesis se enmarca dentro del desarrollo del *broker* ALeRCE (*Automatic Learning for the Rapid Clasification of Events*), el que actualmente procesa en línea el flujo de datos de ZTF y ha sido nombrado *broker* oficial del Rubin Obs.

Tradicionalmente, para la clasificación de objetos astronómicos se utiliza el cálculo de características sobre la curva de luz de los mismos, las que luego son utilizadas para entrenar un clasificador como *Random Forests* (RF [12]). El cálculo de la curva de luz o fotometría [13] para un objeto astronómico, se realiza al computar las imágenes de diferencia entre una imagen inicial (imagen de referencia) del fenómeno y cada una de las imágenes posteriores (imágenes de ciencia) que se toman del objeto, caracterizando sus variaciones de brillo en el tiempo. Algunas de las problemáticas fundamentales que surgen al utilizar curvas de luz es que se necesita de una serie de puntos para caracterizar correctamente el comportamiento de un objeto astronómico y además, se pierde información sobre el resto de la imagen. Tener acceso a la imagen puede resultar provechoso para el proceso de clasificación. Un claro ejemplo de esto, es el sabido hecho de que las supernovas ocurren cerca de una galaxia huésped y

el clasificador podría usar ese conocimiento a-priori para clasificar mejor las supernovas. Se conjetura que este tipo de información, presente en las imágenes de algunos tipos de objetos astronómicos, permite clasificarlos utilizando tan solo las imágenes de la primera observación (primera época), las que corresponden a las imágenes de referencia, ciencia y diferencia. Particularmente, se considera que es posible diferenciar entre estrellas variables, núcleos galácticos activos, asteroides, detecciones erróneas (*bogus*) y supernovas. Se propone un modelo que recibe el nombre de *stamp classifier*, que usando la información de la primera alerta de objetos astronómicos de ZTF es capaz de clasificar entre las 5 clases de objetos astronómicos mencionadas. Una alerta está compuesta por una tripleta de imágenes junto a una serie de metadatos de un objeto en el cielo que ha presentado una variación en su brillo. El trabajo realizado con el stamp classifier cuenta con una publicación aceptada para publicación en el *Astrophysical Journal* [2], donde el autor de esta tesis comparte la autoría principal junto a Rodrigo Carrasco-Davis.

Las primeras noches de observación de un estudio astronómico resultan cruciales, ya que es cuando se observarán objetos nunca antes vistos, pudiéndose descubrir una gran cantidad de fenómenos novedosos y de interés para la ciencia. Por esta y otras razones, se necesita contar con un clasificador temprano. Afortunadamente, es posible obtener una gran cantidad de datos etiquetados al comienzo de los estudios astronómicos, gracias al proceso de *cross-match* [14], lo que significa identificar objetos que ya han sido observados por otros telescopios. De esta manera pueden obtenerse rápidamente etiquetas de estrellas variables, núcleos galácticos activos, asteroides, y supernovas, para entrenar un clasificador temprano de imágenes. Un tipo de detecciones generadas por los estudios astronómicos son los bogus, que ocurren por diferentes motivos; como rayos cósmicos, errores en la medición o anomalías en el proceso de cálculo de imágenes de diferencia, entre otras razones. Las detecciones de tipo bogus son bastante comunes, por lo cual se hace importante agregar este tipo de objetos en los clasificadores utilizados. Sin embargo, no es posible obtener estas detecciones erróneas a partir de *cross-match*, por lo cual un grupo de astrónomos observa una cierta cantidad de detecciones y clasifica a mano cuales de ellas corresponden a bogus, lo que supone no solo un costo en tiempo y recursos humanos, si no que se está perdiendo la capacidad de analizar los objetos de interés en las etapas tempranas del estudio astronómico y además se introduce un posible sesgo. Los objetos *bogus* pueden considerarse como anomalías que se desea evitar, para enfocar recursos y hacer ciencia sobre detecciones que representan fenómenos astrofísicos de interés.

Identificar bogus es necesario para entrenar clasificadores tempranos efectivos, por lo cual se buscan métodos para detectar estas anomalías que pueden definirse como objetos con características diferentes a las detecciones relacionadas a fenómenos astronómicos reales (*inliers*). Se propone trabajar sobre esta problemática desde el punto de detección semi-supervisada de anomalías, donde se cuenta con las etiquetas de los *inliers* astronómicos, obtenidos desde *cross-match*, mientras se desconocen los bogus (anomalías). En la literatura existen variados métodos clásicos para detectar anomalías (*anomalies* o *outliers* en inglés), como *Isolation Forest* (IF [15]), *One-Class Support Vector Machine* (OC-SVM [16]), los que logran buen desempeño sobre datos basados en características. Sin embargo, al momento de aplicarlos sobre imágenes, debido a la maldición de la dimensionalidad [17], su desempeño disminuye. Métodos de aprendizaje profundo (*deep learning*) se presentan como el estado del arte en muchas tareas de visión computacional [18] que involucran datos de alta dimensionalidad,

por lo que se han propuesto muchos algoritmos basados en esta técnica para la detección de anomalías, como los autoencoders convolucionales (*Convolutional AutoEncoders*, CAEs) [19] y las redes adversarias generativas (*Generative Adversarial Networks*, GANs) [20], sin embargo la mayoría no obtiene mejoras significativas al ser comparados con algoritmos clásicos. Recientemente, se han propuesto algoritmos basados en aprendizaje profundo que trabajan sobre transformaciones aplicadas a imágenes, el primero en proponer esta estrategia fue el trabajo *Deep anomaly detection using geometric transformations* [21], donde se aplican una serie de transformaciones geométricas sobre los *inliers* y luego se entrena un clasificador para distinguir entre transformaciones, donde se espera que las transformaciones sobre anomalías sean mal clasificadas. Este método recibe el nombre de *GeoTransform* y ha demostrado funcionar muy bien sobre imágenes al otorgarles un puntaje de normalidad. Posteriormente a GeoTransform, se propuso *Inverse-Transform AutoEncoder* (ITAE [22]), que utiliza un autoencoder para reconstruir una imagen a partir de sus versiones transformadas. Si bien en [22] ITAE se presenta como superior a GeoTransform, sus desempeños son similares y dependen de la base de datos evaluada, por lo que en esta tesis se trabajará principalmente con GeoTransform. Existe mucho espacio para mejorar GeoTransform, ya que no propone explicaciones a su funcionamiento, por ejemplo, no existe claridad de cuales son las transformaciones geométricas más adecuadas para cada base de datos, ya que muchas de estas pueden ser redundantes o simplemente inútiles. Gran parte del trabajo realizado en esta tesis se enfoca en explotar estar posibles mejoras a GeoTransform e incrementar su desempeño en la detección de bogus. Estudios preliminares de proposición de mejoras a GeoTransform han sido publicadas por el tesista como primer autor en el trabajo [4].

Tras implementar mejoras a GeoTransform y contar con el modelo del stamp classifier, se combinan ambas estrategias. Se utiliza GeoTransform para etiquetar bogus de forma automática, y luego estos se utilizan para entrenar el stamp classifier y comparar su desempeño con respecto a utilizar bogus etiquetados por expertos en etapas tempranas o avanzadas del estudio astronómico ZTF. Con esto se busca medir la utilidad de etiquetar bogus con GeoTransform y utilizarlos para entrenar clasificadores en el comienzo de estudios astronómicos, donde no se cuenta con bogus etiquetados por expertos, y de este modo evidenciar si es plausible reemplazar o minimizar la labor de etiquetado manual de bogus por parte de expertos con una estrategia de etiquetado automática que hace uso de GeoTransform.

En la realización de esta tesis se busca entregar herramientas útiles para el *broker* ALeRCE, por lo cual se trabajará mayoritariamente sobre datos del estudio astronómico ZTF. Para realizar pruebas simples y estudiar mejoras sobre algoritmos de detección de anomalías se utilizará una segunda base de datos astronómica denominada *High Cadence Transient Survey (HiTS)*, la que ha sido ampliamente estudiada en trabajos anteriores [23, 24, 25, 26].

Como puede evidenciarse en los párrafos anteriores, este trabajo se divide en tres grandes partes: (i) clasificación temprana de objetos astronómicos, (ii) detección de anomalías tipo bogus, (iii) detección automática de bogus para la clasificación de objetos astronómicos. Cada una estas partes se trata como una sección individual en los capítulos presentados.

## 1.1. Motivación

La clasificación temprana de objetos astronómicos, tiene por objetivo el reportarlos para seguimiento y realizar estudios científicos sobre estos lo antes posible. Tarea que hoy en día es difícil de realizar por seres humanos, debido al volumen de datos generados por los estudios astronómicos, haciéndose necesaria una herramienta de clasificación automática como el stamp classifier.

En el área de detección de anomalías tipo bogus, a pesar de las mejoras propuestas por métodos basados en aprendizaje profundo y transformaciones sobre los datos, se usan conjuntos con transformaciones arbitrarias, no existiendo claridad en cómo seleccionar las transformaciones más útiles para GeoTransform. Su aplicación para detectar bogus surge de la necesidad de identificar este tipo de objetos en etapas tempranas de un estudio astronómico, donde usualmente no se cuenta con muestras etiquetadas como bogus. Además, los objetos de tipo bogus presentan estructuras visuales en sus imágenes que pueden ser resaltadas y aprovechadas por transformaciones específicas.

Finalmente, la detección automática de bogus para la clasificación de objetos astronómicos tiene la motivación principal de eliminar o aligerar la carga de etiquetado manual por parte de astrónomos, y eliminar posibles sesgos.

## 1.2. Hipótesis

En este trabajo se propone la creación de un clasificador temprano de alertas astronómicas, donde se evalua reemplazar al equipo de expertos que clasifica la clase bogus por un algoritmo detector de anomalías, que busca mejorar el estado del arte en datos astronómicos. Según lo mencionado anteriormente, las hipótesis de esta tesis son:

1. En una alerta (imágenes de referencia, ciencia, diferencia y metadatos) de la primera detección de objetos astronómicos existe suficiente información como para entrenar un clasificador capaz de distinguir entre cinco clases: estrellas variables, núcleos galácticos activos, supernovas, asteroides y bogus.
2. Utilizar transformaciones que hagan resaltar las características conocidas a-priori de una muestra tipo bogus permite mejorar los algoritmos de detección de anomalías basados en aprendizaje profundo y transformaciones geométricas (GeoTransform).
3. El mejor conjunto de transformaciones para GeoTransform depende de la base de datos sobre la cual se está trabajando. Por lo cual es posible desarrollar métodos para descartar de forma automática aquellas transformaciones qué no afecten o disminuyan el desempeño de GeoTransform.
4. Un detector de anomalías como GeoTransform es capaz de etiquetar objetos tipo bogus de forma automática, permitiendo entrenar un clasificador de objetos astronómicos, con desempeño similar o superior al que se lograría al utilizar bogus etiquetados por expertos.

## 1.3. Objetivos generales

Desarrollar un clasificador temprano de alertas de objetos astronómicos basado en imágenes y metadatos de la primera detección del objeto, que permita el descubrimiento de nuevas supernovas en el estudio astronómico ZTF. Además se propone desarrollar un algoritmo de detección automática de anomalías (bogus) basado en aprendizaje profundo y transformaciones sobre las imágenes (GeoTransform), el que buscará superar el estado del arte.

## 1.4. Objetivos específicos

Para alcanzar el objetivo general, se han de cumplir con cada uno de los siguientes objetivos específicos:

- O1 Desarrollar un clasificador temprano de objetos astronómicos basado en imágenes y metadatos de la primera detección de alertas del estudio astronómico ZTF.
- O2 Extender GeoTransform para superar el AUROC (*Area Under the Receiver Operating characteristic Curve*) actual del algoritmo, al proponer nuevas transformaciones y métodos para descartar las que disminuyan el desempeño del algoritmo. Además de proponer métodos para encontrar dichas transformaciones.
- O3 Desarrollar una metodología empírica que siente las bases para saber qué transformaciones disminuyen el desempeño de GeoTransform y cuales son redundantes para los algoritmos basados en transformaciones.
- O4 Comparar GeoTransform y sus mejores versiones con otros algoritmos del estado del arte.
- O5 Comparar el desempeño del clasificador temprano al utilizar bogus seleccionados por expertos o los detectados automáticamente por GeoTransform mejorado.

# Capítulo 2

# Marco teórico

En este capítulo se presentan los conceptos importantes que sustentan el trabajo realizado. En un comienzo se aclaran los conceptos relacionados al área de la astronomía, desde el significado mismo de la palabra astronomía hasta la definición de los diferentes tipos de objetos astronómicos existentes. Luego, se explica qué es el aprendizaje profundo (*Deep Learning*), donde se presentan conceptos básicos de redes neuronales y su entrenamiento. Tras esto se presentan conceptos de detección de anomalías donde se define qué es una anomalía y se muestran diferentes algoritmos para su detección. Finalmente, se muestran diferentes aplicaciones de aprendizaje de máquinas en el dominio de la astronomía, desde algoritmos clásicos y de aprendizaje profundo para la clasificación, hasta detección de anomalías en astronomía.

## 2.1. Astronomía

La Real Academia Española define la palabra *astronomía* como *"Ciencia que trata de los astros, de su movimiento y de las leyes que lo rigen."*, y es una ciencia que ha sido estudiada desde los tiempo de Hiparco de Nicea (Nicea, 190 a. de C. - 120 a. de C.) a través de la observación del cielo utilizando la vista desnuda, hasta llegar a los avanzados telescopios robóticos que se usan hoy en día. Los telescopios modernos permiten estudiar objetos celestes a través de diferentes métodos de observación y generan cantidades de datos tan grandes que se hace necesario idear técnicas automáticas para su procesamiento.

El dominio sobre el que se aplican los algoritmos desarrollados en esta tesis es el de las imágenes y metadatos de alertas astronómicas.

### 2.1.1. Estudios astronómicos y big data en la astronomía

Un estudio astronómico o más conocido en inglés como *astronomical survey*, tiene por objetivo generar un mapa general de una región amplia del cielo. A diferencia de la investigación de objetos en particular, donde se utilizan instrumentos para obtener datos específicos, un estudio astronómico puede satisfacer múltiples y variados fines de investigación, siendo parte fundamental de la producción de catálogos astronómicos, donde se recopila la ubicación de objetos astronómicos conocidos y descubiertos por el estudio astronómico.

Los estudios astronómicos usan uno o más instrumentos para recopilar sus datos, por ejemplo un telescopio para recolectar imágenes en diferentes bandas de frecuencia de la luz visible. Para llevar a cabo un estudio astronómico, en primer lugar se define una región del espacio que se desee observar, espectro de luz que se desea observar (frecuencias), instrumentos necesarios para realizar las observaciones, el tiempo que se observará cada región y su cadencia (cada cuanto tiempo se observará cada región del cielo). En base a esto, y considerando las limitantes tecnológicas y monetarias, se construyen los instrumentos para realizar el estudio astronómico. Cuando el instrumento seleccionado para un estudio astronómico es un telescopio, este genera datos crudos en forma de grandes imágenes de las regiones observadas, a partir de las cuales se obtienen *alertas*, gatilladas cuando se tienen variaciones de luminosidad significativa entre dos imágenes de la misma región, y tomadas diferentes instantes de tiempo. Además de imágenes, se generan datos asociados a los parámetros del telescopio, condiciones atmosféricas al momento de la observación, y medidas calculadas a partir de las observaciones, este tipo de datos con magnitudes escalares reciben el nombre de *metadatos*.

Un ejemplo de estudio astronómico es el *Zwicky Transient Facility* (ZTF) *survey* [7], estudio de alta cadencia, donde el cielo observable en el hemisferio norte es capturado por completo cada 3 noches. ZTF utiliza las bandas de luz $g$ y $r$ en sus observaciones, siendo diseñado para ser un estudio astronómico multipropósito, lo que le da la capacidad de impactar una gran variedad de campos de la astronomía. Algunos ejemplos de los muchos estudios astronómicos que se han realizado son HiTS [26], SDSS [27], MACHO [28], ATLAS [29], entre otros.

Uno de los estudios astronómicos más importantes a ser inaugurado en el año 2024 es el *Legacy Survey of Space and Time* (LSST) [6], que va a ser generado con el telescopio Rubin Obs, ubicado en Chile. El estudio astronómico asociado al LSST ha sido diseñado con la región más extensa hasta el momento, capturando todo el cielo observable del hemisferio sur en pocas noches y en 6 bandas de frecuencia de la luz. En la Figura 2.1 se representa con círculos el volumen del cielo observable que son capaces de observar diferentes estudios astronómicos cada vez que el respectivo telescopio realiza una observación del cielo, esta medida recibe el nombre de *etendue*.

En la Figura 2.1 se aprecia cómo LSST es el estudio astronómico con el volumen de observación más grande, donde se estima que se generarán un promedio de 10 millones de alertas por noche. A modo de comparación, el actualmente operativo ZTF genera 1 millón de alertas por noche. Esta cantidad de alertas generadas es clara evidencia de que la astronomía se encuentra en la era del *big data*, donde se genera un cambio de paradigma de observaciones manuales a la necesidad de analizar datos de forma automática. La creación de herramientas automáticas de análisis también permiten estudiar fenómenos en forma temprana, capturando su comportamiento completo, sin perder tiempo para comenzar a hacer ciencia. Como facilitadores de este análisis de datos para la comunidad astronómica es que nacen los llamados *brokers astronómicos* cómo ALeRCE (Automatic Learning for the Rapid Classification of Events) [3], ANTARES (The Arizona-NOAO Temporal Analysis and Response to Events System) [10] o Lasair [11], encargados de desarrollar múltiples herramientas para facilitar que la comunidad astronómica haga ciencia, como modelos de aprendizaje de máquinas o visualizadores de datos.

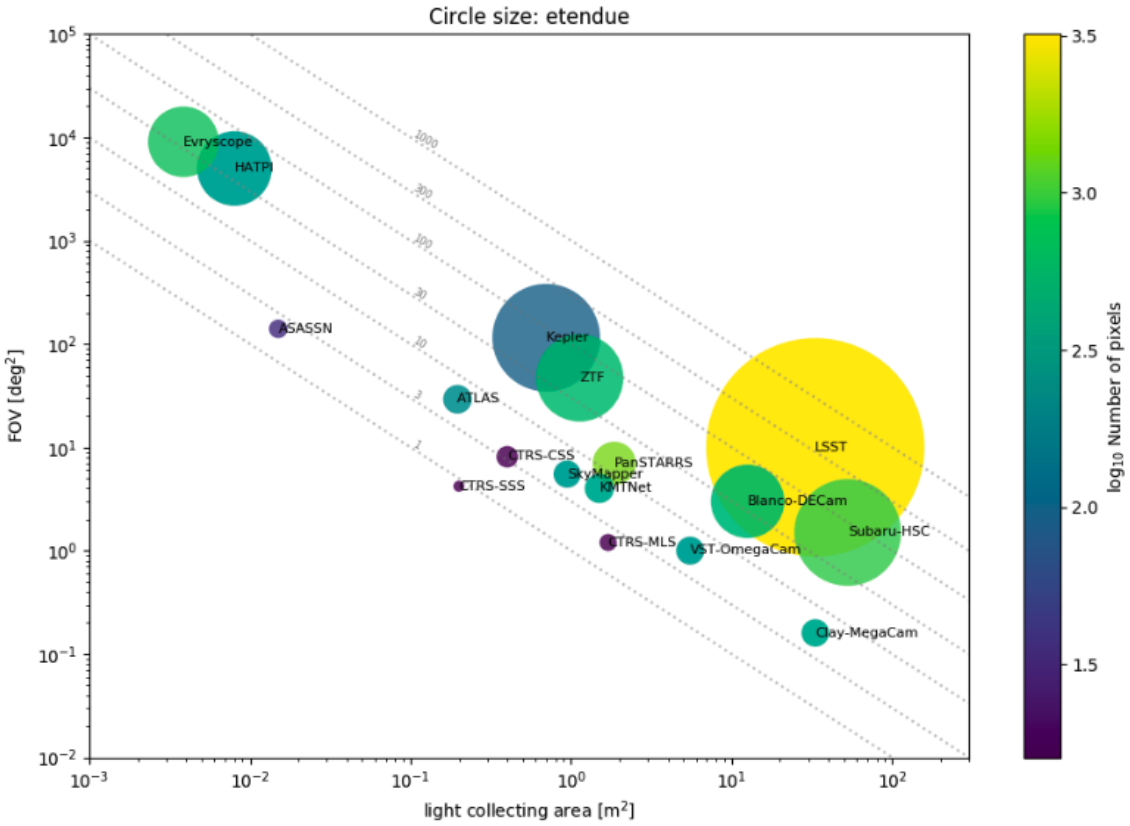



Figura 2.1: Círculos que representan el volumen del espacio observado por cada estudio astronómico (*etendue*) en la observación de una región en particular, i.e., cuando el telescopio se abre y se queda quieto para obtener una imagen. El *etendue* es el producto entre el área de recolección de luz (*light collecting area*, en inglés) en $[m^2]$ y la región de observación (FOV del inglés *field of view*) del telescopio en $[deg^2]$. Figura obtenida de [3].

### 2.1.2. Generación de datos astronómicos

Como se mencionó anteriormente, en una exposición, un telescopio genera una variedad de datos; metadatos, imágenes, alertas, estampas, curvas de luz, etc. A continuación se describe qué compone cada tipo de datos adquiridos y cómo es que son generados.

Una generalización simplificada del proceso de adquisición de datos en una exposición de un telescopio de un estudio astronómico puede verse representado en el flujo de la Figura 2.2. Dado un fenómeno en la región del cielo a la que esta apuntando el telescopio del estudio, la luz de esta región viaja por el espacio y pasa por la atmósfera, viéndose distorsionada por la cantidad de aire que recorre la luz y las condiciones atmosféricas presentes, esta distorsión se caracteriza por una función matemática llamada función puntual de esparcimiento (PSF, por sus siglas en inglés *Point Spread Function*). Al momento de apuntar el telescopio a una región del espacio se escoge un filtro pasa-banda para capturar los fotones de luz en un rango de frecuencias específico. La luz de la región observada es capturada por la cámara del telescopio, la cual está compuesta por CCDs por sus siglas en inglés *charge-coupled devices*. La luz es medida por la CCD en términos de cuentas de fotones, que se traducen en cuentas de electrones, produciéndose la imagen digital de la región del espacio observado. Posteriormente, esta imagen de la región se compara con una imagen de la misma región

pero en etapas tempranas del estudio, y su existen variaciones de luminosidad entre ambas imágenes comienza el proceso de generación de una alerta. Para generar una alerta se centran las fuentes luminosas que generaron la variación de luz y se extraen recortes que contienen a las fuentes, produciendo lo que se denomina "estampas" de la zona donde se registra la diferencia de luminosidad. El recorte de la imagen de la etapa temprana recibe el nombre de imagen de *referencia* (*template* en inglés), mientras que la estampa de la imagen reciente se denomina imagen de *ciencia* (*science* en inglés). Finalmente las imágenes de referencia y ciencia son sometidas a un proceso de substracción (que más complejo que una simple resta y es detallado en [30]) para dar forma a la imagen de diferencia, que en inglés recibe el nombre de *difference*. Esta tripleta de imágenes compone parte de las alertas generadas por el telescopio, las cuales son consideradas como tal sólo si es que las variaciones en la luminosidad estimada presenta una magnitud de luminosidad por sobre los 5-sigma de su error de estimación. Las otras componentes de la alerta son los metadatos asociados a la observación, los que incluyen parámetros del telescopio al momento de la observación, condiciones atmosféricas y otras magnitudes físicas calculadas a partir de la estampas obtenidas.

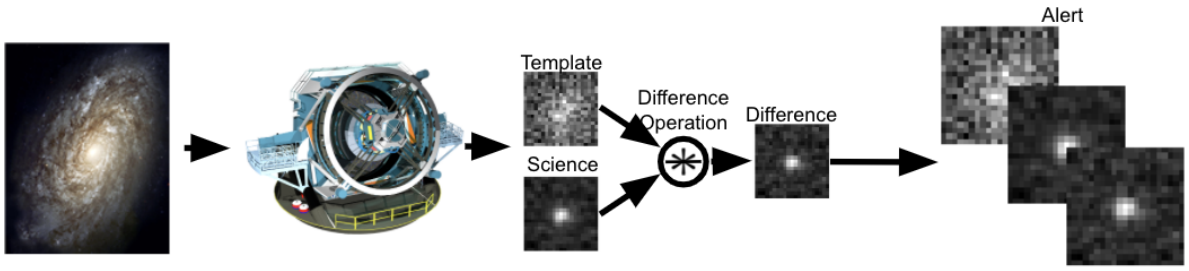



Figura 2.2: Proceso de adquisición de datos en un estudio astronómico. Se genera una variación de luminosidad en una región del espacio, la que es capturada (años después) por un telescopio apuntando a dicha zona y generando una imagen. Luego se compara esta imagen con una de etapas tempranas del estudio astronómico para medir su variación, al restar imágenes recortadas entorno a la región de la variación, con esto se generan la imágenes de referencia, ciencia y diferencia. En esta última se estima la magnitud de luminosidad en la variación, que si está por sobre los 5-sigma de su error de estimación, es considerada una alerta.

En algunas bases de datos generadas a partir de estudios astronómicos se pueden encontrar otros tipos de imágenes que provienen de aplicar procesamientos especiales a alguna de las imágenes originales, e.g. en HiTS se tiene la imagen *SNR difference* (el acrónimo SNR proviene del inglés *signal-to-noise ratio*), compuesta de la imagen de diferencia normalizada por su razón de señal a ruido.

A lo largo de un estudio astronómico es probable que una misma fuente luminosa, con luminosidad variable en el tiempo, genere varias alertas, y en consecuencia varias imágenes de ciencia y diferencia. En vez de caracterizar el fenómeno como una serie de imágenes, se suele calcular la variación de luminosidad presente en cada imagen de diferencia y generar lo que se conoce como curva de luz. Estas curvas son las que se usan típicamente para clasificar los objetos astronómicos en sus diferentes tipos. La magnitud física que mide la variación de

luminosidad presente en la imagen de diferencia recibe el nombre de flujo, y se calcula con un procedimiento llamado fotometría óptima de Naylor [13]. El calculo del flujo posee un error de estimación, si el valor del flujo está 5-sigma por sobre el error de estimación, se considera que la variación de luminosidad no es provocada por el ruido de fondo del cielo, si no que por un objeto astronómico, por lo cual se identifica como una *"alerta"*.

### 2.1.3. Catálogos astronómicos y *cross-match*

Un catálogo astronómico es una compilación, lista o tabulación de una serie de objetos astronómicos, que son agrupados con el objetivo de hacer ciencia y realizar algún tipo de investigación en particular. Los catálogos astronómicos surgen de analizar y estudiar los datos generados por un estudio astronómico particular y los objetos astronómicos que comprenden el catálogo suelen compartir alguna característica en común, el elemento común más usual es el método de adquisición usado por el estudio astronómico, aunque existen catálogos astronómicos compuestos por diferentes fuentes y estudios astronómicos. Algunos ejemplos de catálogos astronómicos son SEDS Messier [31], GSC-II [32], UCAC2 [33], entre otros.

Dado que los catálogos comprenden objetos que son previamente estudiados, son una fuente confiable de etiquetas para las diferentes categorías de objetos astronómicos que existen en el universo. Esto resulta de utilidad para el aprendizaje de máquinas automático, donde se utilizan los catálogos astronómicos como fuentes de etiquetas y datos para el entrenamiento de modelos.

Una estrategia común para la obtención de etiquetas y generación de bases de datos en estudios astronómicos nuevos, donde no se ha creado ningún catálogo aún, es utilizar la información contenida en catálogos astronómicos de otros estudios. De este modo se aprovecha que se tienen registradas las posiciones de los objetos astronómicos etiquetados en los catálogos. Este proceso recibe el nombre de *cross-matching de catálogos astronómicos* [34]. Para las alertas generadas por un estudio astronómico, se busca en una serie de catálogos el objeto con las coordenadas más cercanas para cada alerta, y si estos se encuentran lo suficientemente cerca (distancia definida por quién crea la base de datos), se dice que la alerta es generada por el mismo objeto identificado por el catálogo, y se le asigna la etiqueta respectiva.

El *broker* ALeRCE utiliza extensivamente la técnica de *cross-match* para generar sus bases de datos de entrenamiento, detalles de los catálogos utilizados para este proceso pueden encontrarse en la Sección 3.1.1 de la metodología.

### 2.1.4. Objetos astronómicos

Los objetos astronómicos son definidos como fenómenos naturales o cuerpos físicos que tienen lugar o existen en el universo, y aquellos que emiten o reflejan luz son captados por telescopios en la tierra y en el espacio. Existen muchos tipos de objetos astronómicos como planetas, agujeros negros, estrellas, asteroides, galaxias, etc.

En las bases de datos generadas por ALeRCE se tiene una amplia variedad de objetos astronómicos etiquetados. Los tipos de objetos astronómicos de interés para ALeRCE han sido agrupados en diferentes niveles de una taxonomía jerárquica, un diagrama de esta taxonomía puede verse en la Figura 2.3.

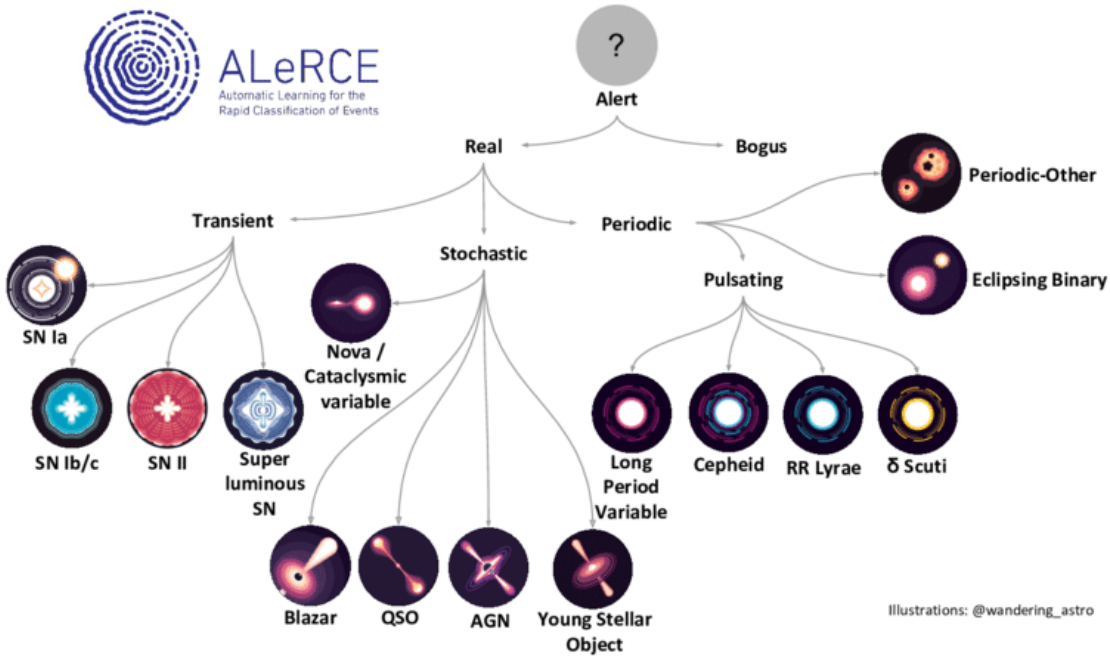



Figura 2.3: Taxonomía utilizada en ALeRCE, se muestra cómo una alerta puede tomar diferentes caminos en un árbol de categorías, donde los nodos son categorías generales que agrupan tipos de objetos que pueden ser definidos de forma cada vez más específica, estos últimos se ubican en las hojas del árbol de la taxonomía. Figura obtenida de [3].

De la taxonomía de la Figura 2.3, para esta tesis resultan relevantes las categorías de real, bogus, *Supernova* (SN) (agrupadas en la categoría *transient* en la Figura 2.3), AGN (del inglés *Active Galactic Nuclei* y en español núcleo galáctico activo) y otras categorías no especificadas en dicha figura; las estrellas variables (VS de sus siglas en inglés *Variable Star*) y asteroides. En este trabajo, las categorías real y bogus se utilizan para los modelos de detección de anomalías y las categorías SN, AGN, VS, asteroide y bogus se utilizan para la clasificación de objetos astronómicos. La razón de utilizar estas categorías en la clasificación dice relación con una de las hipótesis de este trabajo: es posible separar en estas 5 categorías utilizando información de las primeras detecciones como imágenes y metadatos, sin necesidad de la curva de luz. A continuación se presentan definiciones más detalladas para estas categorías relevantes y para las clases utilizadas en clasificación, también se presentan datos relevantes que apoyan la hipótesis de ser distinguibles a partir de imágenes y metadatos:

- **Real**: Los objetos reales son todas las alertas generadas por objetos astronómicos, como se definió al principio de esta sección, los objetos astronómicos son fenómenos o cuerpos que ocurren o existen en el universo. Podrían ser definidos como todo lo que no es del tipo bogus. Algunas investigaciones, como la de esta tesis, consideran que fenómenos como los rayos cósmicos son bogus. Aunque estrictamente hablando los rayos cósmicos son objetos reales, no son objetos de interés astrofísico.
- **Bogus**: Alertas producidas por artefactos en el proceso de adquisición de imágenes o efectos ópticos en la cámara y/o el telescopio, ejemplos de estos efectos son los píxeles saturados al centro de fuentes muy luminosas, columnas de píxeles malos, píxeles

quemados, restas desalineadas en cálculo de la imagen astronómica de diferencia, reflexiones de luz repetitivas al interior del telescopio, etc. Son todas las alertas producidas por fuentes que no son reales ni objetos de interés astrofísico. Dependiendo de la base de datos los bogus pueden incluir objetos astronómicos no deseados como rayos cósmicos o satélites artificiales. Debido a que son artefactos, estos objetos no son incluidos en catálogos astronómicos y la obtención de sus etiquetas suele realizarse por observación humana por parte de expertos. Gran cantidad de las alertas generadas por un telescopio suelen ser del tipo bogus.

- **AGN**: Corresponden a la presencia de agujeros negros en el centro de las galaxias, los que afectan su entorno haciendo que se emita luz. Son objetos astronómicos que varían en su brillo de forma estocástica, las alertas generadas por AGNs presentan un aumento de brillo en la fuente tanto en imágenes de referencia como de ciencia. Al considerar sólo esta característica resulta difícil pensar cómo es que se pueden diferenciar de otros objetos variables, pero los AGNs deberían ubicarse en el centro de sus galaxias anfitrionas (galaxia que alberga al objeto). Cuando la galaxia no es visible, tienden a aparecer en regiones con baja densidad de estrellas visibles.
- **SN**: Las supernovas (SNe, del inglés *Supernovae*) son objetos astronómicos del tipo transientes (fenómenos de duración limitada) y son la explosión de estrellas al final de su ciclo de vida. Una alerta generada por una SN debería manifestarse como un cambio de luminosidad en una ubicación donde no existen fuentes luminosas previamente. Estos objetos suelen ubicarse cercanas a la galaxia anfitriona del fenómeno, con una posición consistente respecto a la población de estrellas implícita en la galaxia (por ejemplo, una SN tiene mayor probabilidad de ubicarse en alineación con el disco de la galaxia que perpendicular a este). De esta forma, la mayoría de las detecciones de SN presentan una galaxia anfitriona visible tanto en las imágenes de ciencia como referencia, y el cambio de luminosidad se percibe sólo en imágenes de ciencia y diferencia. En cuanto a los metadatos, es importante verificar que una SN debería tener un valor de `isdiffpos` 1, para nuevos candidatos a SN, este metadato vale 0 o 1 si la fuente tiene flujo negativo o positivo, respectivamente, en la resta de las imágenes de ciencia y referencia.
- **VS**: Las estrellas variables presentan flujos variables en el tiempo, lo cual puede ocurrir por muchas razones. Un ejemplo de VS son los sistemas de estrellas binarios que al orbitar uno al rededor del otro generan variaciones en la luminosidad de la fuente, otro ejemplo son las estrellas pulsantes cuyo diámetro varía en el tiempo, lo que modifica la intensidad de la luz que emiten. El flujo proveniente de una estrella variable usualmente aparece tanto en la imagen de referencia como en la de ciencia. En telescopios de alta sensibilidad, como el caso de ZTF, las VS son detectadas dentro de la Vía Láctea o del Grupo Local de objetos cercanos a esta, por lo tanto las alertas no deberían presentar imágenes con galaxias anfitrionas. Estas alertas tienen más probabilidad de ubicarse en latitudes galácticas bajas y regiones pobladas de múltiples fuentes, debido a la alta concentración de estrellas en el disco galáctico.
- **Asteroides**: Son alertas generadas por objetos que se mueven dentro del Sistema Solar, usualmente materiales rocosos de diámetros muy variados. Estos objetos deberían aparecer sólo una vez en la posición que son detectados, y presentar flujo sólo en las imágenes de ciencia y diferencia. Dependiendo de su distancia y velocidad aparente, pueden verse como objetos elongados en la dirección del movimiento del asteroide. Además, estas alertas tienen más probabilidad de residir en bajas latitudes eclípticas.

Candidatos a nuevos asteroides siempre deberían tener un valor de `isdiffpos` igual a 1.

### 2.1.5. Alertas astronómicas anómalas

En la adquisición de datos astronómicos existen muchas fuentes de anomalías (la definición de anomalía utilizada en este trabajo se encuentra la Sección 2.3.1), las que, desde el punto de vista astronómico, pueden ser divididas en dos categorías basadas en la naturaleza de su origen:

- **Novedades:** Las novedades que son más conocidas por su nombre en inglés de *novelties*, se definen como anomalías asociadas a objetos de interés para el proceso que se está analizando [35]. En palabras coloquiales, las novedades son anomalías "*deseadas*", las cuales se desea encontrar. En el ámbito de la astronomía, se consideran novedades a aquellas alertas generadas por objetos astronómicos reales de los que se tienen escasos o nulos registros previos, es decir, su descubrimiento presenta la oportunidad de estudiar fenómenos poco estudiados o simplemente desconocidos para la humanidad. Un claro ejemplo de una novedad astronómica es el objeto al que se le dio el nombre de Oumuamua [36], el cual levantó sospechas de tener origen en tecnología alienígena [37], hipótesis que de ser cierta conmocionaría la vida como la conocemos.

- **Artefactos:** Las alertas asociadas a artefactos pueden considerarse anomalías *"opuestas"* a las novedades, son alertas no deseadas presentes en estudios astronómicos, aquellas que no son generadas por objetos astronómicos reales o de interés. En palabras coloquiales pueden considerarse anomalías *"no deseadas"*. Los artefactos astronómicos pueden ser considerados como ruido dentro de un estudio astronómico, el cual se desea filtrar para centrar la investigación en los objetos que realmente interesan. Suelen ser anomalías asociadas al proceso de adquisición y generación de los datos astronómicos. En este trabajo, se considera a los rayos cósmicos y satélites artificiales dentro de la clase bogus, clase que a su vez es considerada como un artefacto, por lo que cada vez que se refiera a un artefacto, en el contexto de astronomía, se estará refiriendo a un bogus, y viceversa. Pese a que en este estudio se considera un artefacto como el equivalente de un bogus, en general esto no es así. Por ejemplo, si se sigue estrictamente la definición de artefacto, objetos astronómicos como los rayos cósmicos o satélites artificiales no son artefactos, sino que objetos astronómicos.

Ambos tipos de anomalías coinciden en que se alejan de las distribución típica de los objetos que se desea estudiar. Este último detalle es importante, ya que no es sólo una desviación de la distribución típica, ya que para el caso de los bogus, estos representan una gran porción de las alertas producidas en un estudio astronómico, lo que se prueba para el estudio ZTF, en la Sección 3.1.7.3, donde se estima que un $\sim$20 % de las alertas generadas por ZTF son bogus, i.e. artefactos.

En lo que respecta a anomalías astronómicas, para este trabajo el principal foco son los artefactos astronómicos de tipo bogus, y no se estudiarán las novedades astronómicas.

## 2.2. Aprendizaje profundo

El presente trabajo se encuentra dominado por la utilización de algoritmos en el estado del arte de aprendizaje de máquinas, los cuales se basan en técnicas de aprendizaje profundo (en inglés *Deep Learning* o DL) [38] en redes neuronales artificiales (ANNs por sus sigla en inglés *Artificial Neural Networks*). No existe un consenso en la literatura de qué significa aprendizaje profundo, pero es común referirse a aprendizaje profundo al entrenamiento de redes neuronales con muchas capas. A continuación se presentan los fundamentos de ANNs, DL y sus aplicaciones sobre clasificación y detección de anomalías.

### 2.2.1. Redes neuronales artificiales

Las ANNs son modelos matemáticos comunmente utilizados para la resolución de problemas de clasificación o regresión. La unidad básica de procesamiento de una ANN recibe el nombre de *neurona* o *perceptrón* [39], esta es una función matemática que recibe un vector de datos $\mathbf{x}$ como entrada, y luego aplica una transformación lineal sobre $\mathbf{x}$, seguida por una función no lineal, la que recibe el nombre de *función de activación*. Cuando se agrupan neuronas se conforma una *capa*, las que reciben el nombre de *capas totalmente conectadas* o, en inglés, *fully connected layers*. La salida producida por un conjunto de $m$ neuronas de una capa *fully connected* se calcula como:

$$\mathbf{y} = \phi(\mathbf{W}\mathbf{x} + \mathbf{b}), \tag{2.1}$$

donde $\mathbf{x} \in \mathbb{R}^n$ es la entrada de la capa, $\mathbf{y} \in \mathbb{R}^m$ es la salida de una capa de $m$ neuronas, $\mathbf{W} \in \mathbb{R}^{m \times n}$ es una matriz de parámetros denominados *pesos*, o *weights* en inglés, $\mathbf{b} \in \mathbb{R}^m$ es un vector que contiene los llamados *sesgos* (*biases* en inglés) de la capa, y $\phi(\cdot)$ es la función de activación no lineal que se aplica tras la transformación lineal de $\mathbf{x}$. Existe una gran variedad de funciones de activación que se pueden escoger para $\phi(\cdot)$, las más utilizadas son las funciones sigmoide (*sigmoid* en inglés), tangente hiperbólica (tanh) y la unidad lineal rectificadora (ReLU, for sus siglas en inglés *Rectified Linear Unit*) [40]. Sus ecuaciones son mostradas a continuación:

$$\begin{aligned} \text{sigmoid}(x) &= \frac{1}{1 + \mathrm{e}^{-x}}, & (2.2) \\ \tanh(x) &= \frac{\mathrm{e}^{x} - \mathrm{e}^{-x}}{\mathrm{e}^{x} + \mathrm{e}^{-x}}, & (2.3) \\ \text{ReLU}(x) &= \text{máx}\{0, x\}. & (2.4) \end{aligned}$$

Para precisar, $\mathbf{W}$ y $\mathbf{b}$ son referidos como los parámetros de la capa *fully connected* de la ANN, y son modificados durante el proceso de entrenamiento del modelo, con el objetivo de ser optimizados para el problema que se busca resolver.

En una ANN, las capas *fully connected* pueden ser agrupadas una después de otra, i.e. de forma secuencial. Por ejemplo, una ANN de dos capas se define cómo:

$$\mathbf{z} = \phi(\mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)}), \tag{2.5}$$
$$\mathbf{y} = \phi(\mathbf{W}^{(2)}\mathbf{z} + \mathbf{b}^{(2)}). \tag{2.6}$$

Los parámetros de la ANN descrita por las ecuaciones 2.5 y 2.6, son $\theta = (\mathbf{W}^{(1)}, \mathbf{b}^{(1)}, \mathbf{W}^{(2)}, \mathbf{b}^{(2)})$, y la ANN en su totalidad puede ser expresada como una función matemática $f_\theta$, que relaciona el vector de entrada $\mathbf{x}$ y de salida $\mathbf{y}$, de la forma $\mathbf{y} = f_\theta(\mathbf{x})$. La forma en que se agrupan las neuronas y capas en una ANN recibe el nombre de *arquitectura* de la ANN.

Cuando las redes neuronales son utilizadas para clasificación, normalmente se utiliza en la última capa del modelo una función de activación que traduzca la salida en probabilidades de clasificación. La función sigmoid$(x)$ cumple esta propiedad, ya que produce salidas acotadas por $(0, 1)$, las que pueden ser interpretadas como la probabilidad de activación para una neurona en la presencia o ausencia de una clase, una propiedad también útil para la clasificación binaria. Una generalización de la función sigmoide, útil para modelos de clasificación multiclase, es la función de activación *softmax*, la que usualmente recibe el nombre de *capa de salida softmax*. Para un problema de clasificación donde se desea encontrar la clase correcta entre un total de $K$ clases, la función *softmax* comprende $K$ neuronas $x_\mathrm{i}$, $\mathrm{i} \in \{1, \ldots, K\}$, donde se desea asignar una probabilidad a cada una, por lo tanto, se requiere que el valor de salida de todas las $K$ neuronas sume 1. Esto se logra al definir la función *softmax* como:

$$\mathrm{softmax}(x_\mathrm{i}) = \frac{\mathrm{e}^{x_\mathrm{i}}}{\sum_{j=1}^{K} \mathrm{e}^{x_j}}, \quad \mathrm{i} \in \{1, \ldots, K\}. \tag{2.7}$$

### 2.2.2. Redes neuronales convolucionales

Las ANNs compuestas exclusivamente de capas *fully connected* se encuentran limitadas a entradas en formato de vectores, además de no tener en consideración la presencia de correlación entre características adyacentes. Para sobreponerse a esta limitación, y preservar el grado de correlación espacial o temporal que pueda existir en la entrada de un modelo, surgen las redes neuronales convolucionales (CNNs, por sus siglas en inglés *Convolutional Neural Networks*). El componente principal de las CNNs son las capas convolucionales, *convolutional layers* en inglés, las cuales mediante una operación de convolución aplican un filtro o kernel a la entrada de la capa. De forma similar a una capa *fully connected*, las salidas de las capas convolucionales son calculadas como sigue::

$$\mathbf{y} = \phi(\mathbf{W} * \mathbf{x} + \mathbf{b}), \tag{2.8}$$

donde $\mathbf{x}$ corresponde a la entrada de la capa, $\mathbf{y}$ es la salida de la capa, $\mathbf{W}$ es el conjunto de filtros a aplicar por convolución sobre la entrada, $\mathbf{b}$ es el vector de sesgos, que posee una componente para cada filtro, y $\phi(\cdot)$ es la función de activación. En este caso, la operación $*$ entre $\mathbf{x}$ y $\mathbf{W}$ es una convolución. Si se considera el caso donde las entradas $\mathbf{x}$ son imágenes, se tiene que $\mathbf{x} \in \mathbb{R}^{\mathrm{e}\times f\times g}$ y $\mathbf{y} \in \mathbb{R}^{u\times v\times l}$ son tensores 3d, mientras $\mathbf{W} \in \mathbb{R}^{\mathrm{d}\times o\times t\times l}$ y $\mathbf{b} \in \mathbb{R}^{l}$. El cálculo de cada elemento $y_{\mathrm{i},j,k}$ de $\mathbf{y}$ se deriva desde la operación de convolución como sigue:

$$y_{\mathrm{i},j,k} = \sum_{m,n,p} x_{\mathrm{i}-m,j-n,p} W_{m,n,p,k} + b_k, \tag{2.9}$$

donde cada elemento $\mathrm{i}, j, k$ del tensor $\mathbf{y}$ se calcula al desplazar los filtros de $\mathbf{W}$ sobre el tensor $\mathbf{x}$ y aplicar la ecuación 2.9. Cada vez que $\mathbf{W}$ se mueve sobre las 2 primeras dimensiones de $\mathbf{x}$, este se salta $S$ píxeles, $S$ recibe el nombre de *stride*. Tras aplicar las capas convolucionales, las primeras dos dimensiones de $\mathbf{y}$ son más pequeñas que las de $\mathbf{x}$. Las dimensiones espaciales (primera y segunda dimensión) del tensor $\mathbf{x}$ y $\mathbf{y}$, se relacionan entre sí como:

$$U = \frac{E - D}{S} + 1, \tag{2.10}$$

donde $U$ es el tamaño de cualquiera de las dimensiones espaciales de $\mathbf{y}$, $E$ es el tamaño de la respectiva dimensión espacial $\mathbf{x}$, $D$ es la respectiva dimensión espacial de $\mathbf{W}$ y $S$ es el *stride* utilizado en la operación de convolución.

Cuando se trabaja en el dominio de las imágenes, usualmente se utilizan filtros cuadrados, con lo que $\mathrm{d} = o$ en las dimensiones de $\mathbf{W} \in \mathbb{R}^{\mathrm{d}\times o\times t\times l}$, y se habla de que la capa tiene $l$ filtros de tamaño d.

En caso de utilizarse dominios temporales, donde las entradas son series de tiempo, las relaciones anteriores resultan equivalentes si se considera la primera dimensión espacial de las variables equivalente a la dimensión temporal, y la segunda dimensión espacial se reemplaza por un valor 1, i.e. $f = o = v = 1$.

Existen muchos otros tipos de capas que componen a las CNNs, incluso, día a día se crean nuevos componentes para estos modelos, ya que son sistemas en el estado del arte del aprendizaje de máquinas. A continuación se muestran algunas de las capas más comunes dentro de las arquitecturas de CNNs, las que también componen los modelos utilizados en esta tesis:

- **Zero-padding:** Es una técnica utilizada para preservar las dimensiones espaciales o temporales de la entrada $\mathbf{x} \in \mathbb{R}^{\mathrm{e}\times f\times g}$ a la salida $\mathbf{y} \in \mathbb{R}^{u\times v\times l}$ de una capa convolucional. El *zero-padding* consiste en agregar 0's a los bordes de las dimensiones espaciales o temporales de la entrada $\mathbf{x}$ de una capa convolucional. Tomando el caso particular de utilizar imágenes como entrada, para una capa convolucional de *stride* $S = 1$, tamaño de kernel $D$, la entrada con *zero-padding* que se alimenta a una capa convolucional ha de tener dimensiones $\mathbf{x} \in \mathbb{R}^{(\mathrm{e}+\lfloor D/2\rfloor)\times(f+\lfloor D/2\rfloor)\times g}$, donde $D/2$ es la cantidad de *zero-padding* incluido para alcanzar una igualdad en las dimensiones espaciales $\mathrm{e} = u \wedge f = v$, entre la entrada de la capa $\mathbf{x}$ y su salida $\mathbf{y}$.
- **Capas de pooling:** Las capas de *pooling* se utilizan en CNNs para reducir la dimensión espacial de sus entradas. Las capas de *pooling* más utilizadas son el *pooling* por máximo (*max pooling* en inglés) y *pooling* por promedio (*average pooling* en inglés). *Max pooling* retorna el valor máximo dentro de una ventana de su entrada $\mathbf{x}$, de forma similar a un filtro convolucional, esta ventana de extracción de valor máximo recorre las dimensiones espaciales (o temporales) de la entrada. La capa de *average pooling* opera de modo

similar a *max pooling*, pero se extrae el promedio de los valores en la ventana. Un caso particular de *average pooling* es el *global average pooling*, donde la ventana utilizada cubre todo el rango de la entrada y la colapsa a un único valor en las dimensiones espaciales (o temporales).

Para ejemplificar el efecto de una capa de *pooling*, se puede considerar el caso típico de CNNs, i.e. la utilización de *max pooling* con una ventana de dimensiones 2×2 y un *stride* de 2, lo que significa que no existe traslape en el recorrido de la ventana por la imagen de entrada. Usualmente, cada vez que se aplica una capa con estos parámetros, se genera una reducción de dimensionalidad espacial a la mitad.

- **Capa de batch normalization:** El *batch normalization* [41] funciona como una capa de normalización entrenable que posee comportamientos diferentes durante el entrenamiento y la evaluación del modelo. Durante el entrenamiento, para cada *batch* (ver Sección 2.2.3 para definición de *batch*) de entrenamiento, esta capa calcula el promedio y varianza de cada característica, para normalizarlas y computar una media móvil exponencial del promedio y la varianza del conjunto de entrenamiento. Una vez finalizado el entrenamiento del modelo, para su evaluación, la población completa de estadísticas ajustadas durante el entrenamiento se utilizan para normalizar las entradas. *Batch normalization* no sólo normaliza los valores de entrada para que tengan un valor promedio cercano a 0 o una desviación estándar cercana a 1, también comprende una ponderación lineal de estas entradas, lo que permite su escalamiento y desplazamiento. Esta capa permite al modelo enfatizar o ignorar características específicas de la entrada, actuando como un regularizador que acelera el proceso de entrenamiento.
- **Dropout:** Es una operación usualmente aplicada a la salida de las capas *fully connected*, aunque recientemente se esta usando sobre capas convolucionales para eliminar filtros completos. *Dropout* se utiliza como un regularizador del modelo para evitar el sobre-ajuste de capas con gran cantidad de neuronas. Similar a la capa de *batch normalization*, *dropout* realiza operaciones diferentes durante entrenamiento y evaluación. La operación de *dropout* se caracteriza por la *tasa de dropout* (*dropout rate* en inglés) $DR \in [0, 1]$, parámetro que, en la fase de entrenamiento del modelo, define la probabilidad de reemplazar cada uno de sus entradas por 0, y multiplicar los valores no reemplazados por 0 por $1/(1-DR)$, de tal forma que la suma sobre todos los valores de entrada se mantenga en el mismo orden de magnitud. En cada paso de entrenamiento un porcentaje $DR$ de las salidas de la capa con *dropout* no serán utilizadas, reduciendo el tamaño efectivo de la capa. Por otro lado, cuando se usa el modelo para evaluar, se desactiva el *dropout*. El efecto deseado de aplicar *dropout* es forzar al modelo a no depender de neuronas específicas en cada capa.

### 2.2.3. Entrenamiento de redes neuronales

El objetivo de utilizar una ANN $f_\theta$ de parámetros $\theta \in \Theta$, es aproximar una función $y = f(x)$, con $x \in \mathcal{X}$. En la práctica no se tiene acceso a la distribución completa de los datos de entrada $\mathcal{X}$, sino que a un subconjunto de $N$ muestras de datos $\{(x^{(i)}, y^{(i)})\}_{i=1}^{N}$ de la función a aproximar, el que recibe el nombre de *conjunto de entrenamiento*. Encontrar el mejor conjunto de parámetros $\theta^*$ para la ANN $f_\theta(x)$ requiere resolver el siguiente problema de optimización:

$$\theta^* = \arg\min_{\theta\in\Theta} \mathcal{C}(\theta) = \arg\min_{\theta\in\Theta} \frac{1}{N}\sum_{\mathrm{i}=1}^{N} \mathcal{L}(y^{(\mathrm{i})}, f_\theta(x^{(\mathrm{i})})), \tag{2.11}$$

donde $\mathcal{C}$ es un funcional de error definido por la función $\mathcal{L}$ que recibe el nombre de **función de pérdida** o **loss function**, en inglés. La optimización mostrada en la ecuación 2.11, se logra a través de técnicas de optimización basadas en **gradiente descendente** (**gradient descent** en inglés), cuando se escoge $\mathcal{L}$ como una función diferenciable (e.g. *entropía cruzada*). Los parámetros $\theta$ son iterativamente ajustados por la siguiente regla, hasta la convergencia:

$$\theta_k = \theta_{k-1} - \mu\nabla_\theta \mathcal{C}(\theta). \tag{2.12}$$

Debido a que las ANNs se componen de muchas capas consecutivas, el cálculo directo de $\nabla_\theta\mathcal{C}(\theta)$ requiere extensivos recursos computacionales. Sin embargo, los gradientes pueden ser eficientemente calculados utilizando la técnica de *retro-propagación* o **back-propagation**, en inglés, algoritmo que *propaga* el error a través del modelo, desde su salida hasta que alcanza la primera capa de la ANN, *back-propagation* se basa en la regla de la cadena para obtener la derivada de funciones.

Incluso cuando se utiliza *back-propagation*, para las redes neuronales entrenadas en grandes cantidades de datos, el cálculo del gradiente exacto $\nabla_\theta\mathcal{C}(\theta)$ se torna computacionalmente costoso. Como solución a este problema, se utiliza una estimación no sesgada del gradiente $\nabla_\theta\tilde{\mathcal{C}}(\theta)$, donde el gradiente es calculado sobre una pequeña porción aleatoria de los datos, porción que recibe el nombre de **batch** y la cantidad de muestras de los datos en un *batch* recibe el nombre de **tamaño del batch** o **batch size** (BS), en inglés. De esta forma, la regla de optimización para un *batch* $\mathcal{B} \subset \mathcal{X}$ es:

$$\theta_k = \theta_{k-1} - \mu\frac{1}{|\mathcal{B}|}\sum_{\mathrm{i}\in\mathcal{B}} \nabla_\theta\mathcal{L}(y^{(\mathrm{i})}, f_\theta(x^{(\mathrm{i})})), \tag{2.13}$$

donde $\mu$ es una constante llamada **tasa de aprendizaje** o **learning rate**, en inglés, y establece cuán grande es el paso de entrenamiento realizado. Esta técnica de entrenar por *batches*, es una forma de *gradiente descendente estocástico* (SGD por sus siglas en inglés *stochastic gradient descent*), y garantiza convergencia cuando $\mu$ es una secuencia bien definida en $k$ que satisface $\sum_k \mu_k = \infty$ y $\sum_k \mu_k^2 < \infty$.

Una alternativa ampliamente utilizada a la regla de optimización de la ecuación 2.13, es Adam [42], un algoritmo de optimización con tasa de aprendizaje adaptiva que ajusta $\mu_k$ de forma automática. Adam utiliza los gradientes elevados al cuadrado para escalar la tasa de aprendizaje e incluye el promedio móvil de los gradientes en su formulación, estrategia que es conocida cómo **momentum**, y es utilizada para evitar la convergencia a un mínimo local durante la optimización. Los principales hiperparámetros de Adam son $\beta_1$ y $\beta_2$, los que se relacionan al promedio móvil de los gradientes y los gradientes al cuadrado, respectivamente, estos regulan la tasa a la que se ajusta la tasa de aprendizaje $\mu_k$.

## 2.3. Detección de anomalías

La detección de anomalías, en el contexto de aprendizaje de máquinas, busca encontrar anomalías dentro de un proceso de adquisición de datos. Sin embargo, definir qué se entiende por anomalía no es tan simple y depende del estudio que se esté realizando. La detección de anomalías es un área de investigación presente en muchos dominios diferentes, ya que siempre que se tiene un proceso de adquisición de datos existirán muestras que se desvíen de lo que se considera normal. Es por esto que surgen técnicas de aprendizaje de máquinas para encontrarlas de forma automática.

A continuación se describen las bases de detección de anomalías sobre las que se construye la investigación de esta tesis, donde se presentan las definiciones de anomalías utilizadas, métodos típicos de aprendizaje de máquinas para la detección de anomalías, y se describen los métodos que inspiraron la creación del modelo GeoTransform, uno de los pilares de este trabajo.

### 2.3.1. ¿Qué es una anomalía?

Como el diccionario *Oxford Languages* la define, una anomalía es *"Algo que se desvía de lo que es estándar, normal o esperado"*, i.e., sólo se puede definir una anomalía a partir de lo que se considera normal. Sin embargo, ¿Qué es algo normal?, la definición de lo que se considera normal es un concepto ambiguo. Este tipo de ambigüedades en la definición de anomalías ha hecho que muchos tipos de problemas en aprendizaje de máquinas sean considerados de detección de anomalías, cada uno acompañado por sus propias definiciones de anomalías y datos normales (en el resto del documento se referirá a los datos normales por su nombre en inglés; *inliers*). Ejemplos de tipos de anomalías según el problema que resuelven son definidos en el siguiente párrafo:

- **Anomalías fuera de distribución:** El nombre en inglés de este enfoque es *Out Of Distribution (OOD) anomaly detection* [43], donde se tiene como *inliers* una base de datos definida y las anomalías son aquellos provenientes de otras bases de datos, es decir, provienen de un dominio completamente diferente, con una distribución de datos distinta. Un ejemplo de esto en astronomía sería utilizar como *inliers* imágenes provenientes de un telescopio y las anomalías son todas las imágenes provenientes desde otros telescopios.
- **Anomalías generales:** Este enfoque es el más típico, y se encuentra al buscar *"detección de anomalías"* en la literatura, en inglés recibe nombre de *"anomaly detection"* [44], *"outlier detection"* [45] o *"novelty detection"* [46]. Para este caso, se considera como anomalía a todo aquella muestra (dato) que sea significativamente diferente a los *inliers*. El significado de *diferente* o *significativamente* queda definido por cada estudio. Este es el tipo de problema estudiado en esta investigación, donde se considera como *diferente* criterios visuales entre imágenes y distancias en distribución de datos, mientras que *significativamente* se propone como todo lo que esté más allá de un umbral fijado a los puntajes de normalidad por el modelo de detección de anomalías utilizado.
- **Anomalías novedosas:** Más conocido por su nombre en inglés de *novelty detection*, es considerado como un subproblema de la detección de anomalías general, donde se buscan anomalías novedosas. Si bien estudios como [46] consideran a este problema

equivalente a la detección de anomalías generales, en áreas como la astronomía, la detección de novedades busca aquellas novedades deseables, tal y como se explicó en la Sección 2.1.5.

- **Artefactos como anomalías:** Considerado un subproblema de la detección de anomalías general. En este caso se definen a las anomalías como aquellos datos que son considerados como ruido respecto a los *inliers*, son datos no deseados. Esto es común en detección de fallas [47] o en la astronomía, cómo se mencionó en la Sección 2.1.5. Este caso particular de detección de anomalías astronómicas es el que estudia este trabajo, donde la anomalías son artefactos del tipo bogus.

En resumen, no existe una definición estándar de anomalía, la que se utiliza en este trabajo es que anomalías son todos aquellos datos significativamente diferentes de los *inliers*, i.e. anomalías generales. Debido a que se considera como anomalías a datos no deseados, en este estudio se trabaja bajo el caso particular de detección de artefactos como anomalías.

### 2.3.2. Funcionamiento de modelos de detección de anomalías

La mayoría de los modelos de aprendizaje de máquinas para detección de anomalías existentes buscan aprender un puntaje de normalidad (o en su defecto, de anormalidad) $n_S(x) : \mathcal{X} \to \mathbb{R}$, que permita mapear el espacio de todas las posibles entradas $\mathcal{X}$ a un valor escalar. De tal forma que mientras más parecida a un *inlier* sea la muestra de entrada $x$, más alto será el puntaje de normalidad $n_S(x)$. Además de este puntaje de normalidad, algunos modelos permiten clasificar la entrada $x$ como anomalía o *inlier*, o también existen métodos para obtener clasificaciones a partir de $n_S(x)$, sin embargo no existe una metodología estándar y resulta ser un problema abierto.

Los modelos que aprenden un puntaje de normalidad $n_S$ son normalmente evaluados en términos de AUROC (del inglés *Area Under the Receiver Operating Characteristic*) o AUPR (del inglés *Area Under the Precision Recall*).

Como se muestra en [44], existen muchos modelos con diferentes estrategias para lograr obtener clasificaciones o un puntaje de normalidad. Según la estrategia utilizada en el aprendizaje los modelos pueden dividirse en aquellos basados en: clasificación de una clase, agrupación (*clustering* en inglés), vecinos más cercanos, aprendizaje estadístico, teoría de la información y análisis espectral.

En lo que respecta a aprendizaje profundo para detección de anomalías, como se muestra en [48], los tipos de modelos pueden ser divididos según la estrategia usada en: AutoEncoders (AE), redes generativas adversarias (GANs, del inglés *Generative Adversarial Networks*), aprendizaje reforzado, redes neuronales de una clase, inferencia variacional, entre otros.

En la Sección 2.3.4 se explican en profundidad los algoritmos de detección de anomalías que son relevantes para esta investigación, ya que el foco no es detallar la extensa cantidad de tipos de algoritmos existentes, las que pueden ser consultadas en múltiples estudios [44, 49, 50, 48, 51].

### 2.3.3. Tipos de problemas de detección de anomalías en base a etiquetas disponibles

Una forma en que se dividen los diferentes problemas de detección de anomalías es el tipo de etiquetas disponibles para entrenar [44], esto modifica las estrategias de entrenamiento, las que pueden dividirse en tres grandes grupos:

- **Supervisada:** La estrategia supervisada, es cuando se tiene acceso a una cierta cantidad de muestras etiquetadas durante el entrenamiento del modelo de detección de anomalías [52]. Algunos trabajos [53] consideran que cuando se utiliza una pequeña cantidad de anomalías etiquetadas durante el entrenamiento y/o hay dos etapas de entrenamiento donde en una se utilizan pocas anomalías, se está usando una estrategia semi-supervisada, cuando en estricto rigor, la utilización de datos etiquetados como anómalos es en realidad una estrategia supervisada. Si se utilizan pocos datos anómalos, puede decirse que es una estrategia débilmente supervisada.
- **No supervisada:** En este caso, se tiene un conjunto de entrenamiento donde anomalías e *inliers* están mezclados y se busca separarlos sin utilizar información de sus etiquetas [54]. Esta es la área que presenta los mayores desafíos, especialmente en casos donde la cantidad de anomalías en los datos es escasa.
- **Semi-supervisada:** En este caso los modelos de detección son entrenados con conjuntos de entrenamiento compuestos únicamente por *inliers*, es decir, se asume que los datos anómalos son desconocidos para los modelos. Este es el enfoque más explorado en la literatura [49, 50, 21] y es conocido como *detección de anomalías por una clase* (*one-class anomaly detection*, en inglés), siendo este el foco de este trabajo.

### 2.3.4. Trabajos relacionado en detección de anomalías

La detección de anomalías es un problema ampliamente estudiado, donde existe una gran cantidad de estudios [49], [50]. La mayoría de los algoritmos de detección de anomalías funcionan bien sobre espacios de características, e.g., cómo *Isolation Forest* (IF) [15] y *One-Class Support Vector Machine* (OC-SVM) [55], pero fallan al intentar detectar anomalías en espacios de alta dimensionalidad, como lo es el dominio de las imágenes. Para solucionar esta problemática, se han propuesto muchas estrategias basadas en aprendizaje profundo, como AutoEncoders (AE) [56], aprendizaje de distribuciones como las GANs [57] y clasificadores entrenados sobre una clase (en inglés llamados *one-class classifiers*) [58].

Recientemente, se han propuesto algoritmos basados en transformaciones aplicadas sobre imágenes, el primero en utilizar este enfoque fue GeoTransform [21], donde una serie de transformaciones geométricas son aplicadas a los *inliers* de entrenamiento. Esto permite crear un conjunto de datos auto-etiquetados compuesto exclusivamente de *inliers*, donde cada transformación tiene su etiqueta propia, con las que se entrena un clasificador para discriminar entre las transformaciones aplicadas a cada muestra. Se espera que cuando se presente una anomalía al clasificador, este no sea capaz de identificar correctamente la transformación aplicada. GeoTransform es uno de los algoritmos centrales a mejorar en esta tesis, por lo cual es explicado en forma separada en la Sección 2.3.5. Desde la publicación de GeoTransform se han creado otros algoritmos basados en transformaciones como Inverse-Transform AutoEncoder (ITAE) [22], que usa un AE para reconstruir imágenes desde sus versiones transformadas.

A continuación, se describe una serie de algoritmos de detección de anomalías basados tanto en aprendizaje de máquinas clásico (enfocados a la utilización de características) como en aprendizaje profundo (principalmente utilizados en el dominio de imágenes). Los métodos descritos son utilizados en la comparación de resudados con GeoTransform y con las mejoras que se proponen.

#### 2.3.4.1. Métodos clásicos de detección de anomalías

**One-Class Support Vector Machine.** OC-SVM se basa en métodos de kernel para la detección de anomalías. Transforma los datos desde el espacio de características a un espacio llamado *Reproducing Kernel Hilbert Space*, donde aprende una frontera de decisión que trata de encerrar todas las muestras *inliers* en un espacio compacto. Todo dato que se encuentre fuera de esta frontera se presume anómalo.

**Isolation Forest.** IF es un algoritmo basado en árboles que funciona por medio de aislar cada muestra con fronteras definidas por árboles de decisión. Mientras menos fronteras de decisión se necesiten para aislar a una muestra más anómala se considera.

#### 2.3.4.2. Métodos de aprendizaje profundo de detección de anomalías

**AutoEncoder Support Vector Machine.** Como un intento de mejora a OC-SVM, en [21] se propone entregar como entrada a OC-SVM la representación aprendida en el cuello de botella de un AE convolucional entrenado sobre imágenes. Para diferenciar este método de OC-SVM, en el resto del documento se refiere a estos métodos RAW-OC-SVM para OC-SVM y CAE-OC-SVM para el modelo que incluye el AE convolucional. Para el AE convolucional, se usa una adaptación del discriminador y generador del modelo llamado *Deep Convolution GAN* (DCGAN) [59], para la red codificadora y decodificadora del AE, respectivamente.

**Deep structured energy-based models.** Modelo basado en aprendizaje profundo, recibe el acrónimo DSEBM [60]. Este modelo funciona al entregar el negativo del logaritmo de la probabilidad (función de energía) asociada a una entrada. Cuando una muestra tiene un valor alto de energía asignado por el modelo, se considera como anómalo. Este modelo utiliza una arquitectura idéntica a la red codificadora de CAE-OC-SVM.

**Anomaly Detection GAN.** ADGAN [61] usa una GAN para aprender el mapeo de una distribución gaussiana multivariada (espacio latente), a la distribución de los *inliers* de entrenamiento. Al momento de evaluación del modelo, se calcula un puntaje de normalidad de una nueva muestra $x$, el cual se computa calculando el error cuadrático medio (MSE, del inglés *Mean Square Error*) entre $x$ y una muestra $G(z)$ generada por la GAN $G(\cdot)$, donde $z$ es un vector proveniente del espacio latente. Luego $z$ se modifica a través de gradiente descendente para minimizar el MSE, y este proceso se realiza 5 veces de forma iterativa. El puntaje de normalidad se define como el MSE entre la muestra original $x$ y la última muestra generada en el proceso iterativo $G(z_5)$. La arquitectura del generador y discriminador de la GAN utilizada en este modelo es la misma que la de las redes codificadora y decodificadora de CAE-OC-SVM, respectivamente.

**Multiple-Objective Generative Adversarial Active Learning.** MO-GAAL [62] es un algoritmo del estado del arte que trata de aprender múltiples GANs sub-óptimas entre-

nadas en *inliers* para poder generar anomalías artificiales. El algoritmo trata de poblar el espacio alrededor de los *inliers* con anomalías generadas y luego se entrena un clasificador para distinguir entre *inliers* y las anomalías artificiales generadas por la GAN, con lo que se espera que las anomalías reales caigan en la misma clase que las anomalías artificiales.

### 2.3.5. Aprendizaje profundo basado en transformaciones geométricas para la detección de anomalías

En el año 2018, se publicó GeoTransform [21], un modelo diseñado para la detección de anomalías en el dominio de las imágenes, el cual se basa en técnicas de aprendizaje auto-supervisado y la utilización de transformaciones geométricas para el diseño de su estrategia de detección de anomalías. GeoTransform fue el primer trabajo en proponer estas técnicas para la detección de anomalías en imágenes y significó un gran avance en el área, al mejorar los algoritmos del estado del arte por un gran margen (diferencias de más de 20 % en métricas como el AUROC, en bases de datos como CIFAR-10 [63]). Se mantuvo como el algoritmo con mejores resultados hasta el 2019, por lo cual ha sido motivo de muchas investigaciones que han propuesto mejorarlo, siendo la inspiración para la proposición de nuevos modelos, como por ejemplo ITAE [22]. Parte de esta tesis busca mejorar y ahondar en el entendimiento del desempeño de GeoTransform.

Como la mayoría de los algoritmos de detección de anomalías, GeoTransform apunta a aprender una función que asigne un puntaje de qué tan normal o parecida a un *inlier* (en inglés *inlier-like*) es una muestra $x \in \mathcal{X}$, donde $x$ se define como un muestra compuesta por un conjunto de imágenes, y $\mathcal{X}$ la distribución completa de muestras de imágenes para un dominio en específico.

En la Figura 2.4 puede verse que para entrenar GeoTransform, en primera instancia, se genera un conjunto de datos *auto-etiquetado* (en inglés *self-labeled*), elemento característico y que le otorga su nombre a las técnicas de aprendizaje auto-supervisado. Este conjunto auto-etiquetado esta compuesto únicamente por *inliers*, donde cada clase corresponde a una transformación geométrica especifica $T_{\mathrm{i}}$, aplicada a todos los *inliers*. El algoritmo original de GeoTransform utiliza un conjunto de transformaciones $\mathcal{T}$ compuesto por una composición de 9 tipos de translaciones espaciales (en inglés *shifts*), 4 tipos de rotaciones, y volteos (en inglés *flips*) verticales de las imágenes en cada muestra, generando un conjunto final de 72 transformaciones $\mathcal{T} = \{T_0, T_1, ..., T_{k-1}\}$, donde $k = 72$. Cada transformación geométrica $T_{\mathrm{i}} : 1 \leq \mathrm{i} \leq k-1$ cambia los valores de los pixeles de las imágenes originales, y $T_0(x) = x$ es la transformación identidad. Al aplicar el conjunto de transformaciones $\mathcal{T}$ a los *inliers* de entrenamiento se genera el conjunto de datos auto-etiquetado $S_{\mathcal{T}}$, cuyas etiquetas corresponden al índice de la transformación aplicada a cada muestra. Debido a que las etiquetas de $S_{\mathcal{T}}$ son generadas de manera artificial y sin inspección visual humana, es que recibe el nombre de conjunto de datos auto-etiquetado. Para cualquier muestra $x \in S$, donde $S$ es el conjunto de datos original (al que no se le han aplicado transformaciones), la etiqueta para la muestra transformada $T_{\mathrm{i}}(x)$ es i. Para culminar el proceso de entrenamiento, se utiliza $S_{\mathcal{T}}$ para entrenar un clasificador $f_\theta$ de $k$ clases y parámetros $\theta$, entrenado a través de la función de pérdida entropía cruzada, para correctamente estimar el índice de la transformación aplicada $T_{\mathrm{i}}$ para cada muestra del conjunto de entrenamiento.

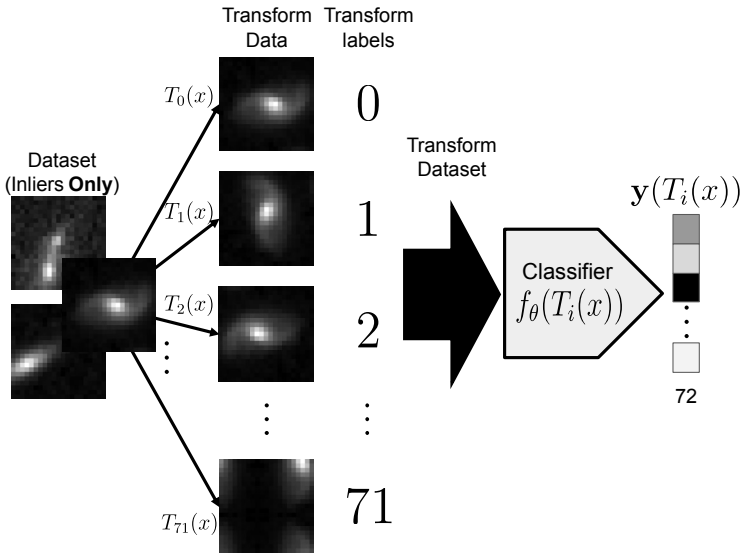
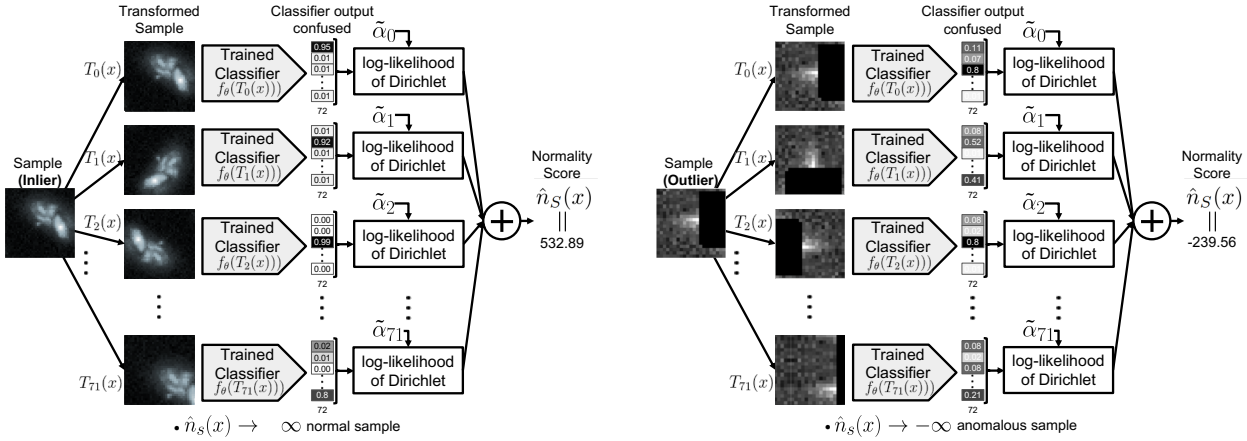



Figura 2.4: Funcionamiento esquemático de la fase de entrenamiento de GeoTransform. Primero se selecciona un conjunto de transformaciones $\mathcal{T} = \{T_0, T_1, ..., T_{k-1}\}$, donde $k = 72$, las que son aplicadas a las muestra del conjunto de datos de entrenamiento con el objetivo de generar un nuevo conjunto de datos auto-etiquetado $S_{\mathcal{T}}$, donde las etiquetas corresponden a los índices de las transformaciones aplicadas. Luego, se entrena un clasificador $f_\theta$ sobre el conjunto de datos auto-etiquetado para estimar los parámetros $\tilde{\boldsymbol{\alpha}}_{\mathrm{i}}$ de una distribución Dirichlet $Dir(\boldsymbol{\alpha}_{\mathrm{i}})$ asociada con las salidas del clasificador, para todas las muestras de entrenamiento transformadas de $S_{\{T_{\mathrm{i}}\}}$, para una transformación dada $T_{\mathrm{i}}$. Figura obtenida de [4].

Como se muestra en la Figura 2.5, para evaluar una muestra $x$, GeoTransform usa un *puntaje de normalidad Dirichlet* (en inglés *Dirichlet normality score*) $n_S(x)$, el que es definido como la combinación de la log-verosimilitud del vector de salida *softmax* $\mathbf{y}(x) \triangleq softmax(f_\theta(x))$ proveniente del clasificador $f_\theta$, para cada muestra transformada $T_{\mathrm{i}}(x)$, condicionada en la i-ésima transformación aplicada $T_{\mathrm{i}}$, tal y como se representa en la siguiente ecuación:

$$n_S(x) \triangleq \sum_{\mathrm{i}=0}^{k-1} \log p(\mathbf{y}(T(x))|T = T_{\mathrm{i}}). \tag{2.14}$$

El nombre Dirichlet en este puntaje de normalidad, proviene de que se utiliza el supuesto que todas las distribuciones condicionales $p(\mathbf{y}(T(x))|T = T_{\mathrm{i}})$ son independientes entre sí y siguen una distribución Dirichlet, entonces se cumple $\mathbf{y}(T(x))|T = T_{\mathrm{i}} \sim Dir(\boldsymbol{\alpha}_{\mathrm{i}})$, donde $\boldsymbol{\alpha}_{\mathrm{i}} \in \mathbb{R}_+^k$, $x \sim p_X(x)$, $\mathrm{i} \sim Uni(0, k-1)$, y $p_X(x)$ es la distribución de datos de las muestras *inliers*. El vector de parámetros de máxima verosimilitud $\boldsymbol{\alpha}_{\mathrm{i}}$, de la distribución Dirichlet $Dir(\boldsymbol{\alpha}_{\mathrm{i}})$ para cada transformación $T_{\mathrm{i}}$, necesita ser calculado, pero encontrar los valores del vector de parámetros $\boldsymbol{\alpha}_{\mathrm{i}}$ es un problema intratable (en inglés *intractable*), así que este es estimado a través de métodos numéricos [64, 65]. La estimación es representada por $\tilde{\boldsymbol{\alpha}}_{\mathrm{i}}$, con lo que el puntaje de normalidad puede escribirse como:




(a) (b)

Figura 2.5: Funcionamiento esquemático de fase de evaluación de GeoTransform. En ambas figuras se muestra cómo se usa GeoTransform para evaluar una nueva muestra $x$, donde se le aplican todas las transformaciones y luego se obtienen las salidas del clasificador. Para cada salida, se calcula la log-verosimilitud de la muestra al usar el parámetro Dirichlet $\tilde{\boldsymbol{\alpha}}_{\mathrm{i}}$ asociado a la respectiva transformación i. Luego, todas las log-verosimilitudes son sumadas para dar con el puntaje de normalidad $n_S(x)$. Mientras más negativo es el puntaje obtenido, más anómala se considera a la muestra evaluada. En (a) la muestra es un *inlier* y el puntaje de normalidad es alto. Por otro lado, en (b) la muestra es una anomalía y se obtiene un puntaje de normalidad bajo, ya que el clasificador no es capaz de discriminar correctamente entre las transformaciones aplicadas.

$$n_S(x) = \sum_{\mathrm{i}=0}^{k-1} \left[ \log \Gamma(\sum_{j=0}^{k-1} \tilde{\boldsymbol{\alpha}}_{\mathrm{i}j}) - \sum_{j=0}^{k-1} \log \Gamma(\tilde{\boldsymbol{\alpha}}_{\mathrm{i}j}) + \sum_{j=0}^{k-1} (\tilde{\boldsymbol{\alpha}}_{\mathrm{i}j} - 1) \log \mathbf{y}(T_{\mathrm{i}}(x))_j \right], \tag{2.15}$$

donde $\Gamma(\cdot)$ corresponde a la función Gamma y la expresión con tres términos entre paréntesis de corchetes es una forma de escribir la log-verosimilitud de una distribución Dirichlet sujeta al vector de parámetros $\tilde{\boldsymbol{\alpha}}$, extraída desde [65]. Tomando en consideración que los vectores de parámetros $\tilde{\boldsymbol{\alpha}}_{\mathrm{i}}$ son constantes respecto a $x$, los primeros dos términos dentro de los paréntesis de corchetes pueden ser ignorados sin afectar el ordenamiento de normalidad otorgado originalmente por el puntaje $n_S$, de este modo se puede definir un puntaje de normalidad simplificado:

$$n_S(x) = \sum_{\mathrm{i}=0}^{k-1} \sum_{j=0}^{k-1} (\tilde{\boldsymbol{\alpha}}_{\mathrm{i}j} - 1) \log \mathbf{y}(T_{\mathrm{i}}(x))_j = \sum_{\mathrm{i}=0}^{k-1} (\tilde{\boldsymbol{\alpha}}_{\mathrm{i}} - 1) \cdot \log \mathbf{y}(T_{\mathrm{i}}(x)). \tag{2.16}$$

El proceso descrito en los párrafos anteriores corresponde a la fase de entrenamiento del algoritmo GeoTransform, ilustrado en la Figura 2.4. Para evaluar este modelo y computar el puntaje de normalidad $n_S(x)$ sobre una nueva muestra, se han de aplicar todas las transformaciones de $\mathcal{T}$ a la muestra a evaluar, y luego calcular la salida *softmax* del clasificador, para

cada una de las versiones transformadas de la muestra a evaluar. Por cada vector *softmax* $\mathbf{y}(T_i(x))$, se computa la log-verosimilitud usando el respectivo vector de parámetros $\tilde{\boldsymbol{\alpha}}_i$, previamente ajustado a los datos de entrenamiento, y finalmente se suman las log-verosimilitudes de todas las transformaciones para obtener $n_S(x)$. Este proceso se ilustra en la Figura 2.5. Particularmente, la Figura 2.5a muestra la respuesta del modelo al evaluar una muestra *inlier*, donde las transformaciones aplicadas son correctamente identificadas por el clasificador, obteniéndose altas probabilidades en los índices del vector *softmax* correspondiente a la transformación aplicada, lo que otorga un valor alto de puntaje de normalidad $n_S$. Por otro lado, en la Figura 2.5b se presenta al modelo una muestra anómala, donde el modelo erra en muchos casos al tratar de identificar las transformaciones aplicadas, lo que genera valores bajos de log-verosimilitud y un puntaje de normalidad que tiende a menos infinito $n_S \to -\infty$.

#### 2.3.5.1. Arquitectura de GeoTransform

GeoTransform utiliza un clasificador $f_\theta$ para identificar las diferentes transformaciones aplicadas a una muestra, en principio, $f_\theta$ puede ser cualquier clasificador, pero ya que se está trabajando en el dominio de las imágenes se utilizan CNNs. Siguiendo los lineamientos del algoritmo original de GeoTransform [21], se utiliza como clasificador una Wide Residual Network (WRN) [66] con parámetros de profundidad 10 (número de capas convolucionales) y factor de ensanchamiento 4 (multiplicador de cantidad de filtros en bloque residual, respecto al número original de filtros, que se duplica tras cada bloque). En forma detallada, la arquitectura de la WRN descrita consiste de 7 capas convolucionales y 3 conexiones residuales, en una configuración mostrada en la Figura 2.6.

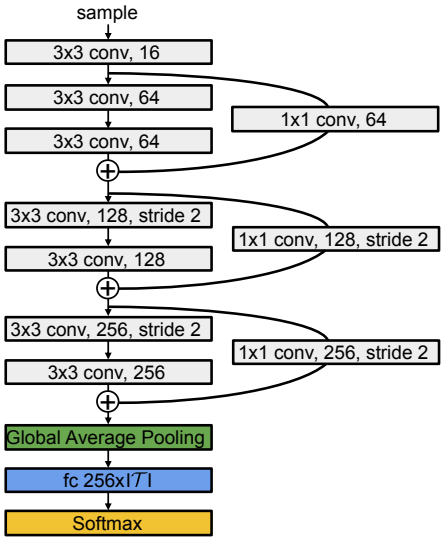



Figura 2.6: Arquitectura del clasificador Wide Residual Network utilizada por GeoTransform, la cual es construida con parámetros de profundidad 10 y factor de ensanchamiento 4, incluye 7 capas convolucionales, 3 conexiones residuales, un *average pooling* global y una capa *fully connected* con tamaño de salida igual al número de transformaciones aplicadas $|\mathcal{T}|$. Al principio y final de cada conexión residual, se aplica *batch normalization* con momentum 0.9 y $\varepsilon = 1^{-5}$. Figura obtenida de [4].

#### 2.3.5.2. Entrenamiento de GeoTransform

GeoTransform [21] originalmente usa un tamaño de *batch* de 128, optimizador Adam y entropía cruzada como función de pérdida. GeoTransform suele ser entrenado durante $[200/|\mathcal{T}|] = [200/72] = [2.\bar{7}] = 3$ épocas sobre el conjunto de datos auto-etiquetado $S_{\mathcal{T}}$. Este esquema de entrenamiento se adopta con el objetivo de imitar el mismo número de iteraciones que se lograrían si es que el clasificador utilizado fuese entrenado sobre el conjunto de datos original $S$ (sin transformaciones) por 200 épocas. Este esquema se adopta para comparar GeoTransform con otros algoritmos de detección de anomalías basados en aprendizaje profundo, los cuales en [21] son entrenados durante 200 épocas sobre el conjunto de datos original $S$.

### 2.3.6. Desde aprendizaje auto-supervisado hasta GeoTransform y más allá

El funcionamiento de GeoTransform está basado en el aprendizaje auto-supervisado (SSL de las siglas en inglés *self-supervised learning*) [67], donde se genera un conjunto de datos *auto-etiquetado*. Además, para la detección de anomalías GeoTransform se vale del fenómeno que cuando se presentan muestras fuera del dominio de los datos de entrenamiento de una red neuronal, las salidas de esta son impredecibles. Pese a esto, en [21] no se muestra literatura relacionada a ninguno de estos conceptos, pareciendo que fuera una idea que surge de la nada, sin embargo existen investigaciones previas que estudian estos conceptos de tal forma que pueden ser considerados como precursores o modelos que sentaron las bases para GeoTransform. A continuación se presenta un breve contexto de los conceptos clave de GeoTransform, los modelos precursores a GeoTransform y finalmente un modelo que se aprovecha de estos conceptos para mejorar GeoTranform.

La técnica de **aprendizaje auto-supervisado** se refiere a todo proceso de aprendizaje donde la tarea de supervisión es creada de forma artificial, yendo desde la creación de etiquetas para ordenar imágenes separadas en piezas de rompecabezas [68], hasta regeneración de agujeros creados en imágenes [69], entre otros. El entrenamiento en estas tareas auto-supervisadas, suele realizarse para que el modelo aprenda características de los datos, las que puedan ser útiles para mejorar el desempeño en otras tareas. La investigación del área con relación más estrecha a GeoTransform es el trabajo de [70], donde se entrena un clasificador de forma auto-supervisada sobre un conjunto de datos con imágenes y etiquetas correspondientes a rotaciones de las imágenes originales. Luego, a este clasificador se le hace un ajuste fino para clasificar el conjunto de datos original en categorías de los objetos que contiene. Ésta es la primera investigación donde se tiene registro del uso de transformaciones geométricas para tareas auto-supervisadas. De forma similar a [70], en GeoTransform también se usan rotaciones para crear el conjunto de datos auto-supervisado, agregando la novedad de componer la rotación con otras transformaciones geométricas. Actualmente, las técnica de aprendizaje auto-supervisado se basan principalmente en la utilización de funciones de costo *contrastivas* (en inglés, *contrastive*) [71] que pretenden separar muestras muy diferentes en el espacio de características, para lo que utilizan modelos basados en energía (en inglés, *energy based models*) [72].

El fenómeno de **salidas impredecibles** en clasificadores a los que se presentan datos no

presentes en el conjunto de entrenamiento, ha sido estudiado en la literatura de la detección de anomalías del tipo *fuera de distribución*, tipo de anomalías que fueron definidas y discutidas en la Sección 2.3.1 de este marco teórico. En [43] se muestra que cuando se presentan muestras de un dominio diferente al de los datos en que fue entrenado un clasificador, este presenta probabilidades con mayor entropía y menor certeza en la probabilidad escogida, por lo cual utilizan el máximo de la salida *softmax* del clasificador como puntaje de anomalía en las muestras, puntaje que recibe el nombre de *maximum softmax score*. Al mezclar un clasificador entrenado de forma auto-supervisada sobre transformaciones geométricas con la idea de que las anomalías generan salidas de alta entropía, se obtiene GeoTransform.

De forma posterior al trabajo de *maximum softmax score*, se presenta *outlier exposure* (OE) [73], donde se ahonda en el comportamiento de las salidas impredecibles y se propone incentivar que muestras fuera de la distribución de entrenamiento generen salidas de alta entropía en el clasificador. Para esto se presentan muestras de otras clases o bases de datos diferentes durante el entrenamiento de un clasificador y se optimiza la entropía cruzada para un vector de distribución uniforme, de esta forma, cuando el clasificador observe muestras fuera de la distribución de entrenamiento debería generar salidas *softmax* con distribuciones que tiendan a la uniforme. Al momento de entrenar con OE se tiene cuidado de que las anomalías usadas durante el entrenamientos sean diferentes a las usadas para evaluar el modelo. Una característica de OE es que puede ser implementada sobre cualquier clasificador basado en ANNs. Esto es explotado en [74], donde se mejora un modelo muy similar a GeoTransform, donde se entrena un clasificador de forma auto-supervisada sobre transformaciones geométrica, y además en el proceso de entrenamiento se muestran datos de una distribución diferente a la del conjunto de entrenamiento. Se optimiza el modelo para que su salida sea uniforme en estos casos. Este modelo reporta un AUROC promedio de 95.6 % en la base de datos CIFAR-10, frente al AUROC de 82.3 % obtenido por GeoTransform.

## 2.4. Trabajo relacionado de aprendizaje de máquinas en astronomía

### 2.4.1. Aprendizaje de máquinas clásico para clasificación en astronomía

En este trabajo cuando se habla de aprendizaje de máquinas clásicos, se refiere a todo lo que no es aprendizaje profundo, es decir, modelos que funcionan en su mayoría sobre el procesamiento de características, donde el proceso de de ingeniería de características (idear nuevas características) es fundamental. Ejemplos de estos modelos son los perceptrones multicapa [39], máquinas de soporte vectorial (SVM del inglés *Support Vector Machine*) [75], bosques aleatorios (RF del inglés *Random Forest*) [76], entre otros.

Desde el advenimiento de la era del big data en la astronomía, la utilización de algoritmos de aprendizaje de máquinas ha ganado fuerza, debido a la necesidad de procesar masivas cantidades de información. La mayoría de los esfuerzos se han concentrado en la utilización de técnicas clásicas basadas en el perfeccionamiento de características basadas en el conocimiento científico en torno a la astronomía.

La estrategia más común en la clasificación de objetos astronómicos es la utilización de métodos basados en árboles, como RF, aplicados sobre características extraídas a partir de las curvas de luz de objetos astronómicos. El proceso de cálculo de características consiste en tomar las curvas de luz, y después calcular características sobre estas como; período, media, desviación estándar, etc. Para calcular características usualmente se usan librerías como FATS (*Feature Analysis for Time Series*) [77]. Esta técnica es utilizada hasta el día de hoy, un ejemplo de esto es ALeRCE, donde utiliza este modelo para clasificar objetos en el estudio astronómico de ZTF [1].

### 2.4.2. Aprendizaje profundo para clasificación en astronomía

La literatura del aprendizaje profundo en astronomía suele estudiar los problemas desde dos enfoques; la utilización de curvas de luz para la clasificación de objetos astronómicos [78, 79] y la utilización de imágenes de alertas astronómicas. Los trabajos previos en clasificación utilizando imágenes astronómicas se han concentrado en clasificación de morfología de galaxias [80, 81, 82], clasificación de objetos de series de imágenes en el tiempo [24, 83], y en su mayoría, en la detección de alertas de objetos reales y bogus [84, 85, 25, 23, 86, 87]. Estos últimos algoritmos son usados para el filtrado automático de bogus, utilizando un enfoque de entrenamiento supervisado, donde las alertas de tipo bogus suelen ser manualmente etiquetadas por grupos de expertos.

El modelo de clasificación de objetos astronómicos utilizado en este trabajo se basa en Enhanced Deep-HiTS [23], donde se utiliza una CNN para clasificar imágenes entre objetos reales y bogus. En este modelo se introduce invarianza rotacional total a las imágenes de entrada, lo que se logra al alimentar la CNN con versiones rotadas en 90°, 180° y 270° de las imágenes originales, para finalmente promediar las representaciones intermedias generadas para cada rotación por las capas densas de la CNN, operación que recibe el nombre *cyclic average pooling*.

### 2.4.3. Detección de anomalías astronómicas

La mayoría de los trabajos relacionados a detección de anomalías utilizan modelos clásicos de aprendizaje de máquinas; en [88] se usa IF y otras técnicas de detección de anomalías clásicas sobre características calculadas desde curvas de luz de SNe, en [89] se calculan características sobre imágenes de galaxias para entrenar un IF. Si bien se pueden encontrar trabajos donde se use aprendizaje profundo, como en [90], sólo se usa aprendizaje profundo para la extracción de características de imágenes astronómicas de galaxias, las que luego se usan para entrenar métodos clásicos como IF. De forma similar a este algoritmo y como primer trabajo que busca detectar anomalías del tipo bogus se tiene [91], donde se extraen características sobre imágenes de objetos reales para entrenar un IF, usando características diseñadas a mano, en vez de utilizar imágenes directamente. El único trabajo que utiliza estrategias basadas de forma exclusiva en aprendizaje profundo sobre imágenes astronómicas es [92], donde se entrena una GAN para detectar imágenes de galaxias anómalas.

# Capítulo 3

# Metodología

En esta sección se describe la metodología llevada acabo para cumplir los objetivos del trabajo de tesis. Como se mencionó en la Sección 1 de la Introducción, el trabajo realizado se divide en tres grandes partes:

(i) **Clasificación temprana de objetos astronómicos**: Se usa un modelo que recibe el nombre de stamp classifier.

(ii) **Detección de anomalías tipo bogus**: Se proponen mejoras a GeoTransform para detectar bogus.

(iii) **Detección automática de bogus para la clasificación de objetos astronómicos**: Utilización de GeoTransform para etiquetar bogus automáticamente y mejorar el desempeño del stamp classifier en etapas tempranas de un estudio astronómico.

Es por esta división que la metodología presentada a continuación también se divide en explicar el trabajo realizado en estas tres partes.

Tanto la parte (i) como la parte (ii) del trabajo cuentan con publicaciones asociadas ([2] para la parte (i) y [4] para la parte (ii)), en [4] el tesista es el único autor principal y en [2] se comparte la autoría principal con Rodrigo Carrasco-Davis, donde el tesista tuvo responsabilidad total en el desarrollo de modelos, y obtención de resultados para mejorar el desempeño del stamp classifier (cuando se muestre un resultado no obtenido por el tesista esto se menciona de forma explícita), participando de forma colaborativa en la escritura del artículo publicado. En esta tesis, se presentan los resultados obtenidos por el tesista en ambas publicaciones, los cuales son extendidos, particularmente en las partes (ii) y (iii). Cabe mencionar que el modelo stamp classifier se encuentra en operación dentro del proyecto ALeRCE, y está en mejora continua.

## 3.1. Clasificación temprana de objetos astronómicos

A continuación se describe el trabajo realizado entorno al punto (i) mencionado anteriormente, i.e., resolver el problema de clasificación temprana de 5 tipos de objetos astronómicos: VS, AGN, SN, asteroides y bogus. Esta tarea se resuelve a través de la implementación de un

clasificador basado en CNNs, el cual recibe el nombre de *stamp classifier*. El stamp classifier utiliza como entradas la información de alertas astronómicas, compuestas por imágenes y metadatos de primeras detecciones del flujo de alertas generadas por el estudio astronómico ZTF. La utilización de primeras alertas se debe a que el principal objetivo del stamp classifier es la temprana detección de SNe, de este modo se pueden estimar probabilidades de objetos apenas se reciben las alertas.

El stamp classifier ha sido ampliamente validado y descrito a fondo en [2]. Debido a la amplia extensión de los experimentos y resultados mostrados en [2], los menos relevantes para esta tesis se mencionan de forma superficial. Para más detalles revisar [2].

### 3.1.1. Base de datos

Para entrenar el stamp classifier se utiliza una base de datos de alertas de ZTF, construida por el equipo de ALeRCE el 6 de mayo del 2020. En el resto del documento se refiere a esta base de datos con el nombre de ALeRCE-ZTFv5. Esta base de datos contiene alertas de primeras detecciones, compuestas por imágenes y metadatos que contienen toda la información relevante relativa a la fuente que gatilla la alerta [93]. Cada alerta contiene 3 imágenes denominadas estampas, a las que por simplicidad se referirá como imágenes en el resto del documento. Cada imagen es un recorte de $63 \times 63$ pixeles, obtenida de una imagen original que contiene una gran región del cielo. Dicho recorte contiene en su centro la fuente que gatilla la alerta, cada píxel equivale a 1 arco-segundo de declinación en el cielo. Además, cada alerta contiene metadatos relacionados con la fuente, las condiciones de observación de la toma del telescopio y otra información útil [94]. Una lista completa y descripción de los metadatos disponible en cada alerta puede encontrarse en `https://zwickytransientfacility.github.io/ztf-avro-alert/schema.html`.

Los objetos astronómicos de categoría real (AGN, SN y VS) de la base de datos ALeRCE-ZTFv5 son un subconjunto extraído de una base de datos más grande (más objetos y múltiples alertas por objeto) construida por el equipo de ALeRCE [1], la que es el resultado de *cross-match* con el catálogo ASAS-SN de estrellas variables [95, 96, 97, 98], el catálogo Roma-BZCAT Multi-Frequency de Blazars [99], el catálogo Million Quasars (versión de Junio del 2019) [100, 101], el New Catalog of Type 1 AGNs (Oh2015) [102], los Catalina Surveys Variable Star Catalogs [103, 104], el catálogo LINEAR de curvas de luz periódicas [105], el catálogo *Gaia* Data Release 2 [106, 107], la base de datos SIMBAD [108], y SNe espectroscópicamente clasificadas de la base de datos de *Transient Name Server* (TNS). Por otro lado, la clase asteroide de la base de datos ALeRCE-ZTFv5 se construye al seleccionar las alertas que fueron gatilladas cerca de asteroides conocidos dentro del sistema solar. Las alertas de tipo bogus en la base de datos ALeRCE-ZTFv5, se construyeron en 2 etapas de etiquetado manual, en primer lugar se tienen los *bogus del paso 1*, compuestos por 1980 alertas de bogus reportadas por ZTF (en base a inspección humana), con los que se entrena una versión inicial del stamp classifier. Luego, se incluyen los *bogus del paso 2*, donde otras 8783 alertas fueron etiquetadas como bogus por un equipo de expertos de ALeRCE usando el *SN Hunter* (detallado más adelante, en la Sección 3.1.8) e incluidos en la base de datos al inspeccionar manualmente las alertas predichas como SNe por versiones iniciales del stamp classifier.

El número total de alertas en la base de datos ALeRCE-ZTFv5 es de 52,244, las cuales se

dividen en 14,966 (29 %) AGN, 1620 (3 %) SNe, 14,996 (29 %) VS, 9899 (19 %) asteroides, y 10,763 (20 %) bogus. El subconjunto de alertas reales obtenidas de [1] contiene esta cantidad de alertas con el objetivo de contar con un mejor balance entre las clases. En la Figura 3.1 pueden observarse varias tripletas de imágenes que componen cada una de las clases de la base de datos.

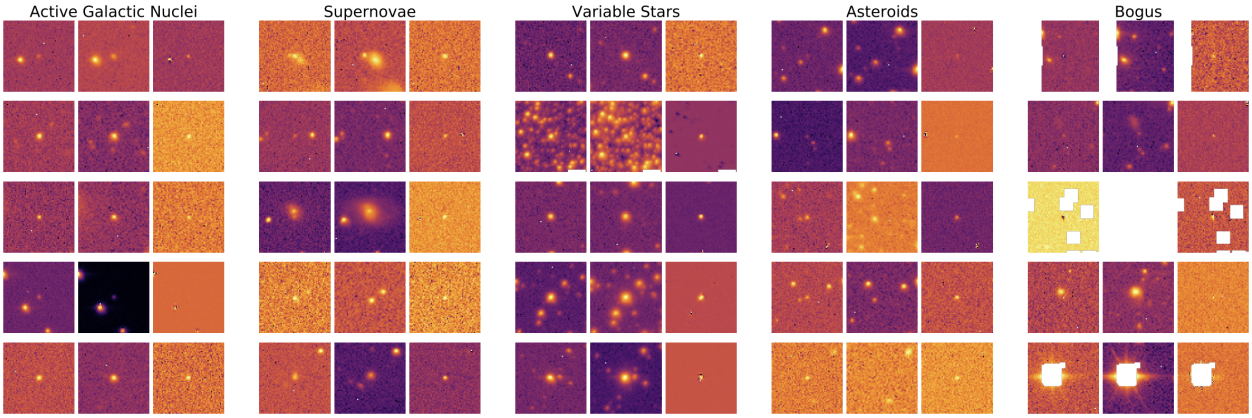



Figura 3.1: Ejemplos de las tripletas de imágenes contenidas en alertas de primeras detecciones de la base de datos ALeRCE-ZTFv5, se muestran imágenes para cada una de las 5 clases presentes en la base de datos. Para cada clase las 3 imágenes en cada fila son las imágenes de ciencia, referencia y diferencia, de izquierda a derecha. Las que corresponden a una alerta. Figura obtenida de [2].

### 3.1.2. Pre-procesamiento

El flujo de pre-procesamiento para las imágenes y metadatos de las alertas que entran al stamp classifier se resume a continuación:

- **Imágenes:**
  1. Descartar alertas con imágenes no cuadradas.
  2. Recortar imágenes a 21×21 pixeles.
  3. Normalizar cada imagen de una alerta para que tenga valores entre $[0, 1]$, sin considerar pixeles con valores NaN (no numéricos o *Not a Number*, en inglés).
  4. Reemplazar pixeles NaN por el valor 0.
- **Metadatos:**
  1. Se escogen los 23 metadatos detallados en la Tabla A.1 del Apéndice.
  2. Se limitan los valores máximos y mínimos a los metadatos y valores de la Tabla A.2 del Apéndice.
  3. Se normalizan los metadatos para que distribuyan $\mathcal{N}(0, 1)$, calculando los estadísticos de media ($\mu$) y desviación estándar ($\sigma$) sobre el conjunto de entrenamiento de metadatos.

A continuación se explica esto en más detalle, así como los efectos del flujo de cada elemento del pre-procesamiento:

- **Imágenes:** El tamaño estándar para las imágenes de una alerta es 63×63 pixeles, sin embargo, existen ocasiones en que se generan alertas en los bordes de las grandes imágenes generadas por la CCD del telescopio, lo que produce imágenes no cuadradas. Para mantener esta consistencia se remueven 650 alertas con imágenes de tamaños no cuadrados de la base de datos ALeRCE-ZTFv5. Tras remover alertas con imágenes no cuadradas, se obtienen 14,742 (29 %) AGN, 1596 (3 %) SN, 14,723 (29 %) VS, 9799 (19 %) asteroides y 10,734 (20 %) alertas bogus, con un total de 51,594 alertas.

  Pruebas preliminares han mostrado que imágenes más pequeñas generan mejores resultados de clasificación, por lo tanto se recortan todas las imágenes al centro para lograr imágenes de tamaño 21×21 pixeles; este tamaño se selecciona en base a una búsqueda aleatoria de hiperparámetros, la cual es discutida en la Sección 3.1.7.1.

  Algunos pixeles de las imágenes, particularmente de la clase bogus, contienen valores NaN, debido a la saturación de pixeles por fuentes muy brillantes o malas columnas; todos los pixeles NaN se reemplazan con el valor 0, entregando al clasificador información sobre el contenido de NaNs en cada imagen.

  Cada imagen es normalizada de forma independiente entre 0 y 1 al restar el valor mínimo de los pixeles en la imagen y luego dividiendo por el máximo de este resultado. Finalmente, de las imágenes de cada alerta se arma un cubo de 3 canales, el cual se utiliza como entrada para el clasificador, este cubo se construye al apilar en canales separados las imágenes de ciencia, referencia y diferencia resultantes del pre-procesamiento ya descrito, generándose un tensor de tamaño $21 \times 21 \times 3$.
- **Metadatos:** Otra entrada del clasificador es un vector con 23 características (especificados en la Tabla A.1 del Apéndice) asociados a cada alerta, vector que en el resto del documento será referido como vector de metadatos, este vector es preprocesado al limitar cada metadato en un intervalo de valores definido por la Tabla A.2 del Apéndice (este proceso recibe el nombre en inglés de *clipping*). Luego, cada metadato es normalizado al substraer $\mu$ y dividiendo por $\sigma$ para que distribuya $\mathcal{N}(0, 1)$, estos estadísticos se calculan sobre metadatos del conjunto de entrenamiento.

### 3.1.3. Partición de datos

Debido al desbalance de clases y a que el pre-procesamiento remueve imágenes no cuadradas, la partición de datos se realiza después del pre-procesamiento. Tanto para el conjunto de validación como el de test se utilizan 100 alertas por clase, es decir, ambos conjuntos quedan con 500 alertas cada uno, mientras que el conjunto de entrenamiento queda con el resto de las 50,594 alertas. La decisión de utilizar 100 alertas por clase en los conjuntos de test y validación se basa en mantener la mayor cantidad de alertas de la clase SN dentro del conjunto de entrenamiento, mientras se mantienen conjuntos de test y validación balanceados con suficientes alertas para validar el correcto funcionamiento del modelo de clasificación propuesto. Se busca tener al menos 1000 imágenes por clase en el conjunto de entrenamiento, en base a la regla de dedo [109] originada a partir de la base de datos ImageNet [110], donde hay aproximadamente 1000 imágenes por cada clase.

Antes de entrenar el stamp classifier, se realiza un balanceo de las clases del conjunto de entrenamiento por sobre-muestreo, donde se replican las alertas de las clases menos representadas hasta que alcancen el mismo número de alertas que la clase con mayor número de

alertas, es decir, todas las clases del conjunto de entrenamiento quedan con 14,542 alertas, con lo que se tiene un conjunto de entrenamiento con un total de 72,710 alertas.

### 3.1.4. Arquitectura del clasificador

El modelo de clasificación utilizado para el stamp classifier es una CNN basada en el clasificador real-bogus propuesto en [23], el cual es una mejora sobre Deep-HiTS [25] al agregar invarianza rotacional total a la CNN y analizar las predicciones del modelo utilizando Layer-wise Relevance Propagation (LRP) [111]. La arquitectura específica de la CNN utilizada en este trabajo se muestra en la Tabla 3.1. En previos trabajos de la literatura no se incluyen metadatos como entrada para los clasificadores.

Las entradas del stamp classifier están compuestas por la tripleta de imágenes que componen las alertas de ZTF: imágenes de ciencia, referencia y diferencia, las que se pre-procesan a una forma de $21 \times 21 \times 3$, como se explica en la Sección 3.1.2, además de un vector de 23 metadatos. La parte convolucional del modelo recibe un *batch* con las imágenes de cada alerta y siguiendo la arquitectura planteada en [23], se aplica un *zero-padding* (rellenar los bordes de las imágenes con ceros) a la entrada, para luego aumentar el *batch* con versiones rotadas de sí mismo en 0°, 90°, 180° y 270° (si el *batch* de entrada es $x$, y $r$ es la operación de rotación en 90°, entonces el *batch* de entrada aumentado queda como $B(x) = [x, rx, r^2x, r^3x]$). Para las capas convolucionales, se utilizan los parámetros mostrados en la Tabla 3.1, donde se detallan las dimensiones de los filtros y el número de canales de salida. Todas las capas convolucionales, excepto la primera, incluyen *zero-padding* que preserva la forma de la entrada tras aplicar la convolución. Además, todas las capas convolucionales y *fully connected* tienen ReLU por función de activación (exceptuando la última capa *fully connected* que tienen una salida *softmax*). La salida de la última capa de *max pooling* es aplanada para formar los vectores que alimentan las capas *fully connected*. Luego de la primera capa *fully connected*, se produce la concatenación de las rotaciones, donde se apilan las representaciones vectoriales de las versiones rotadas de cada alerta, las que son alimentadas a la capa de *average cyclic pooling*, donde se aplica un promediado en la dimensión apilada. Como entrada paralela al modelo, se procesa el vector de metadatos con una capa de *batch normalization*, los que luego son concatenadas a la salida de la capa de *average cyclic pooling*. Estas representaciones concatenadas pasan a través de dos capas *fully connected*, para finalmente aplicar una función *softmax* a la salida del clasificador y obtener las probabilidades estimadas por cada una de las 5 clases a predecir. Una explicación más detallada de CNNs, sus capas y forma de entrenamiento, puede ser consultada en la Sección 2.2 del Marco Teórico.

Una representación gráfica de la arquitectura del stamp classifier puede verse en la Figura 3.2. En esta figura se aprecia cómo es que se aplican las rotaciones a las imágenes de entrada, para lograr la invarianza rotacional. La utilización de esta técnica se justifica en que, normalmente, los objetos astronómicos presentes en una imagen presentan orientaciones aleatorias y se ha mostrado en la literatura que imponer invarianza rotacional a clasificadores tipo CNNs, mejora su *accuracy* para algunos problemas de clasificación [80, 112, 25, 23].

Tabla 3.1: Arquitectura de red neuronal convolucional del stamp classifier.

| Capa | Parámetros de la capa | Tamaño de salida |
|---|---|---|
| Entrada | - | $21 \times 21 \times 3$ |
| *Zero-padding* | - | $27 \times 27 \times 3$ |
| Aumentación por rotaciones | - | $27 \times 27 \times 3$ |
| Convolución | $4 \times 4, 32$ | $24 \times 24 \times 32$ |
| Convolución | $3 \times 3, 32$ | $24 \times 24 \times 32$ |
| *Max pooling* | $2 \times 2$, *stride* 2 | $12 \times 12 \times 32$ |
| Convolución | $3 \times 3, 64$ | $12 \times 12 \times 64$ |
| Convolución | $3 \times 3, 64$ | $12 \times 12 \times 64$ |
| Convolución | $3 \times 3, 64$ | $12 \times 12 \times 64$ |
| *Max pooling* | $2 \times 2$, *stride* 2, | $6 \times 6 \times 64$ |
| Aplanado | - | 2304 |
| *Fully connected* | $2304 \times 64$ | 64 |
| Concatenación de rotaciones | - | $4 \times 64$ |
| Cyclic pooling | - | 64 |
| Concatenación con metadatos tras BN[a] | - | $64 + 23$ |
| *Fully connected* con *dropout* | $90 \times 64$ | 64 |
| *Fully connected* | $64 \times 64$ | 64 |
| Salida *softmax* | $64 \times 5$ | 5 (n° clases) |

[a] BN significa *batch normalization*

### 3.1.5. Regularización de entropía

Cuando se entrena el stamp classifier utilizando entropía cruzada como función de costo a minimizar, la confidencia de clasificación del modelo resulta ser muy alta, generándose una salida con una distribución de probabilidades saturada en valores de 0s y 1s, sin un rango intermedio, incluso cuando se producen clasificaciones erróneas. En este caso se pierde la percepción de confidencia (el valor de las probabilidades relativas entre clases) de las predicciones, esto por que la mayoría de las probabilidades estimadas para cada clase resultan ser 0 o 1. Con el objetivo de entregar más granularidad a las predicciones, y entregar mayor información a los astrónomos que verifican candidatos a SN, es que se agrega la entropía de las probabilidades predichas por el modelo como término regularizador en la función de costo. Término que se maximiza durante el entrenamiento. Segun investigaciones [113] al maximizar la entropía de las probabilidades de salida, se logra penalizar aquellas predicciones con alta confidencia, lo que permite realizar un mejor análisis en los casos donde la alerta parece pertenecer a más de una clase con la misma certeza. La función de costo $\mathcal{L}$ utilizada para implementar la mencionada regularización, es la siguiente:

$$\mathcal{L} = \underbrace{-\sum_{c=1}^{N} y_c \log\left(\hat{y}_c\right)}_{\text{entropía cruzada}} + \underbrace{\beta \sum_{c=1}^{N} \hat{y}_c \log\left(\hat{y}_c\right)}_{\text{regularización de entropía}} , \tag{3.1}$$

donde $N$ es el número de clases, $y_c$ es la etiqueta de la alerta en codificación *one-hot* (un

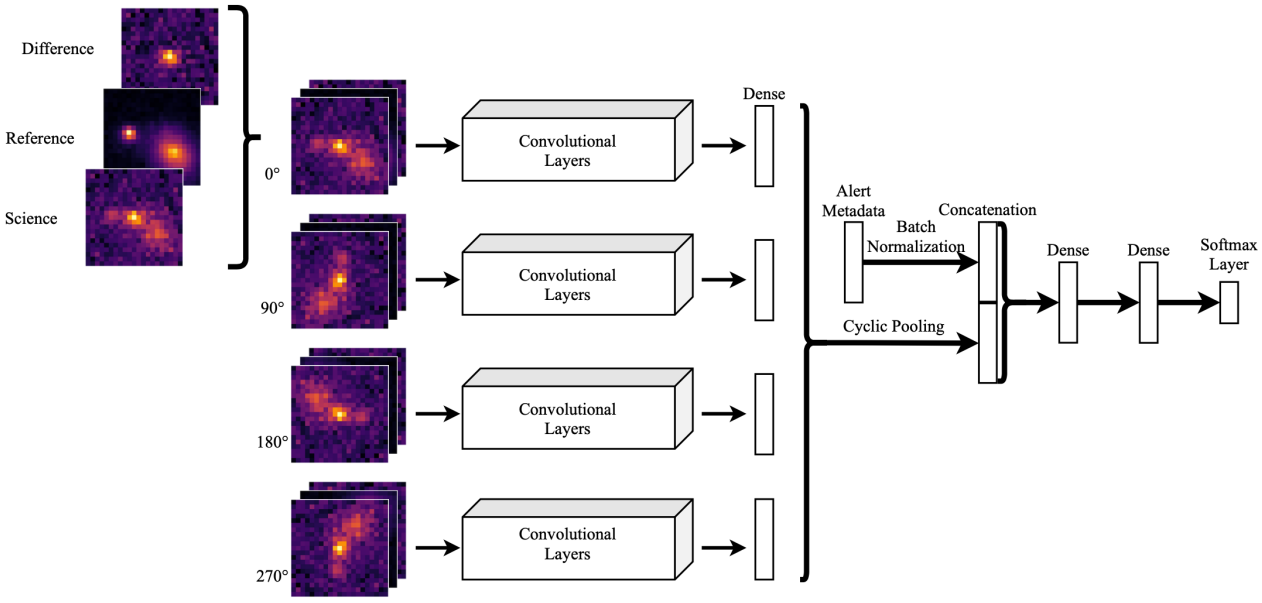



Figura 3.2: Modelo del stamp classifier, compuesto por una CNN mejorada con invarianza rotacional. Las cajas indicando *Convolutional Layers* se refieren a las descritas en la Tabla 3.1, desde la primera capa convolucional hasta la última de *pooling*. Cada alerta a la entrada está compuesta por un tensor de dimensiones $21 \times 21 \times 3$ que contiene las imágenes de ciencia, referencia y diferencia de una alerta. Por cada tripleta de imágenes se generan versiones rotadas, las que son alimentadas a la CNN. Después de la primera capa *fully connected*, se realiza la operación de *cyclic average pooling*. El vector de metadatos de la alerta de entrada se pasa por una capa de *batch normalization*, y su salida es concatenada con la salida de la capa de *average cyclic pooling*. Luego, esta concatenación se pasa por 2 capas *fully connected*, para finalmente pasar por una función *softmax* que entrega las probabilidades estimadas para cada alerta. Figura obtenida de [2].

vector de largo $N$ que contiene un valor de 1 en el índice $c$ de la clase de la alerta y 0 para el resto), $\hat{y_c}$ es la predicción del modelo para la clase $c$, y $\beta$ es una constante que controla la influencia del término de regularización de entropía respecto al término de entropía cruzada en la función de costo.

### 3.1.6. Esquema de entrenamiento

El entrenamiento del stamp classifier utiliza un máximo inicial de 30,000 iteraciones de entrenamiento (en cada iteración se presenta un único *batch*), donde se evalúa la función de costo en el conjunto de validación cada 10 iteraciones para guardar el mejor modelo hasta el momento. Luego de las primeras 20,000 iteraciones, si es que se encuentra un menor valor de la función de costo en el conjunto de validación, se realizan 10,000 iteraciones más. Los conjuntos de validación y test se muestrean de forma aleatoria una única vez al comienzo de la investigación, al tomar 100 alertas por clase desde la base de datos ALeRCE-ZTFv5 completa, con lo que se obtienen 500 alertas tanto para el conjunto de validación como el de test. Estos conjuntos se mantienen fijos a lo largo de todos los experimentos del stamp classifier y el resto de las alertas son utilizadas en el conjunto de entrenamiento.

Con el objetivo de que los *batch* en cada iteración del entrenamiento contengan, aproximadamente, el mismo número de alertas por clase, es que antes de comenzar el entrenamiento se realiza un balanceo de las clases del conjunto de entrenamiento por sobre-muestreo, donde se replican las alertas de las clases menos representadas hasta que alcancen el mismo número de alertas que la clase con mayor número de alertas, es decir, todas las clases del conjunto de entrenamiento quedan con 14,542 alertas.

Se utiliza Adam [42] como regla de optimización para actualizar los parámetros del modelo en cada iteración del entrenamiento, con parámetros $\beta_1 = 0.5$ y $\beta_2 = 0.9$.

Para ajustar los hiperparámetros de entrenamiento del stamp classifier se utilizan los conjuntos de entrenamiento y validación de la base de datos ALeRCE-ZTFv5, al realizar una búsqueda aleatoria de hiperparámetros la que se detalla en la siguiente Sección 3.1.7.1.

## 3.1.7. Experimentos

A continuación se describe la metodología para realizar experimentos para evaluar el stamp classifier. Como se ha mencionado, el stamp classifier cuenta con una publicación [2], donde parte de la investigación se desarrolla en la colaboración del proyecto ALeRCE. Por lo cual en la siguiente sección se detallan los experimentos más relevantes para esta tesis, desarrollados de forma individual por el autor de este trabajo y en la secciones 3.1.7.3 y 3.1.8 se mencionan una serie de experimentos extra, y sus detalles pueden ser consultados en el artículo publicado del stamp classifier [2].

### 3.1.7.1. Búsqueda de hiperparámetros

Para encontrar la mejor combinación de hiperparámetros del modelo stamp classifier para la base de datos ALeRCE-ZTFv5, es que se realiza una búsqueda al muestrear al azar 133 combinaciones de los parámetros mostrados en la Tabla 3.2. Para cada combinación de hiperparámetros, se entrenan 5 redes con pesos iniciales diferentes e inicializados de forma aleatoria.

Tabla 3.2: Valores para búsqueda aleatoria de hiperparámetros.

| Hiperparámetro (nombre en inglés) | Valores de búsqueda aleatoria |
|---|---|
| Tasa de aprendizaje (*learning rate* LR) | 5e-3; 1e-3; 5e-4; 1e-4; 5e-5 |
| Parámetro de regularización (*regularization param.* $\beta$) | 0; 0.3; 0.5; 0.8; 1.0 |
| Tamaño de *batch* (*batch size* BS) | 16; 32; 64 |
| Tamaño de imágenes de entrada (*image size* IS) | 21; 41; 63 |
| Tasa de *dropout* (*Dropout rate* DR) | 0.2; 0.5; 0.8 |
| Tamaño de kernel de la CNN (*kernel size* KS) | 3; 5; 7 |

### 3.1.7.2. Desempeño y evaluación del modelo

Se analiza el desempeño del modelo stamp classifier en la clasificación de 5 clases al medir el *accuracy* y matriz de confusión promedio de 5 entrenamientos del modelo. Se utilizan los mejores hiperparámetros encontrados en la búsqueda aleatoria de hiperparámetros. También

se muestran alertas correctamente e incorrectamente clasificadas por el modelo en el conjunto de test.

Para evidenciar el efecto que tiene la utilización de metadatos en el desempeño del modelo, es que se entrena una versión del stamp classifier sin utilizar metadatos, es decir, sólo haciendo uso de las imágenes de una alerta como entrada al modelo y se compara respecto al modelo entrenado con metadatos.

Finalmente, se evidencian los efectos de entrenar el stamp classifier con diferentes valores de regularización de entropía $\beta$, al analizar cómo varían las probabilidades predichas para las diferentes clases de objetos astronómicos.

#### 3.1.7.3. Experimentos extra

A continuación se describe una serie de experimentos realizados con el stamp classifier, donde se hace énfasis en validar sus resultados sobre la detección de SNe y bogus, tanto en la base de datos ALeRCE-ZTFv5 como en el flujo de alertas generadas cada noche por ZTF. Los resultados y detalles de estos experimentos extra pueden encontrarse en el artículo publicado del stamp classifier [2]. Son dos las razones por la cual estos experimentos no se incluyen en esta tesis; primero, algunos resultados fueron realizados por múltiples integrantes del proyecto ALeRCE y no se puede atribuir total autoría al autor de esta tesis, los experimentos de esta lista que si hayan sido realizados exclusivamente por el tesista se indican con un **[Autoría exclusiva]** al comienzo. Segundo, los experimentos extra no son de relevancia para este trabajo y son considerados como apéndices de este trabajo.

La lista de experimentos extra que se puede encontrar en el artículo publicado del stamp classifier [2] incluye:

- Evaluación de desempeño del stamp classifier para la detección de las SNe de la base de datos ALeRCE-ZTFv5, donde se mide *recall* y curva ROC.
- Validación de qué tan buenas son las predicciones realizadas por el stamp classifier en un conjunto de datos de 390,498 primeras alertas no etiquetadas, las que se obtuvieron del flujo de alertas de ZTF, de tal forma de que sus coordenadas estuvieran aproximadamente uniformemente distribuidas en el cielo. Se realiza un análisis de la la distribución en el cielo de las diferentes clases predichas por el stamp clasifier para el conjunto de datos no etiquetado. Para cada clase se verifica la correcta ubicación espacial, ya que ciertas clases presentan coordenadas preferentes, e.g., asteroides deberían encontrarse cerca de la eclíptica (camino que sigue el Sol en el cielo, observado desde la Tierra), AGNs y SNe deberían encontrarse en mayor concentración fuera del plano galáctico, etc.
- **[Autoría exclusiva]** Estudio de cuales de los metadatos resultan más relevantes para las predicciones del stamp classifier. Se realiza un *ranking* de las características más importantes con un clasificador RF entrenado sobre los metadatos. Luego, se mide el *accuracy* del stamp classifier agregando de uno en uno los metadatos, desde el más relevante al menos relevante, según RF.
- **[Autoría exclusiva]** Se mide cual de las 3 imágenes de las alertas astronómicas es más importante para la clasificación. Se calcula el *accuracy* del stamp classifier entrenándolo

en todas las posibles combinaciones de imágenes.

- **[Autoría exclusiva]** Utilizando el stamp classifier se estima la proporción de bogus en las alertas generadas por el flujo de alertas de ZTF. Se estima que ∼20 % de todas las alertas generadas son bogus, y que ∼60 % de estas tienen una sola detección.
- **[Autoría exclusiva]** Se analizan en profundidad los tipos de bogus presentes en la base de datos ALeRCE-ZTFv5. Se definen los tipos de bogus observados por los expertos de ALeRCE en el stream de alertas de ZTF. Los bogus de la base de datos ALeRCE-ZTFv5 se proyectan en un U-MAP [114] y se visualiza si es que las agrupaciones encontradas en U-MAP coinciden con las diferentes definiciones de tipos de bogus. Se visualizan las diferencias de las proyecciones de U-MAP; los *bogus del paso 1*, los *bogus del paso 2* y las alertas de SNe, donde se observa una fuerte similaridad en estas dos últimas, debido a que los *bogus del paso 2* son alertas incorrectamente etiquetadas como SNe por versiones iniciales del stamp classifier.
- Se validan las correctas predicciones del stamp classifier respecto al clasificador de curvas de luz de ALeRCE (LCC, del inglés *light-curve classier*) [1], sobre el conjunto de 390,498 alertas no etiquetadas, sólo se utilizan aquellos objetos que poseen 6 o más detecciones, condición necesaria para ser evaluados por el LCC.

### 3.1.8. Supernovae Hunter - Validación del modelo por la comunidad.

El stamp classifier es un modelo utilizado por el equipo de ALeRCE de forma continua. Para su fácil utilización por astrónomos, ingenieros de esta colaboración desarrollaron una herramienta llamada *SN Hunter* que toma el stamp classifier, y pone a disposición en tiempo real sus predicciones sobre el flujo de alertas de ZTF. Gracias a esta herramienta, que tiene como bloque principal el stamp classifier desarrollado en esta tesis, los astrónomos de ALeRCE han sido capaces de **descubrir más de 6,846 nuevas SNe** las que han sido espectroscópicamente verificadas por TNS, en un periodo de tan sólo un año de operación. Posicionando a ALeRCE como broker competitivo a nivel mundial en la identificación de SNe, respecto al resto de las iniciativas que analizan el flujo de alertas de ZTF.

Detalles de cómo funciona el SN Hunter y cómo es que lo utilizan los astrónomos de ALeRCE para descubrir nuevas SNe, pueden ser consultados en [2], donde también se muestran diferentes análisis que comparan la capacidad de ALeRCE para reportar SNe, en comparación con otras iniciativas.

## 3.2. Detección de anomalías tipo bogus

En esta sección se busca describir la metodología llevada a cabo para mejorar el algoritmo de detección de anomalías llamado GeoTransform [21] en su aplicación sobre la detección de anomalías astronómicas, las que reciben el nombre de *bogus* a lo largo del documento. Parte del trabajo presentado en esta sección fue publicado por el tesista en [4], el cual es extendido, con una profundización sobre la selección de transformaciones.

GeoTransform es entrenado utilizando objetos astronómicos reales, los que serán referidos como *inliers* en el resto de esta sección. GeoTransform cumple el objetivo de detectar

anomalías de tipo bogus de forma automática y aliviar la tarea de etiquetado manual a astrónomos.

A diferencia de la sección anterior, en esta sección, los datos de entrada del modelo son referidos como muestras, ya que están conformados sólo por las imágenes astronómicas (ciencia, referencia y diferencia) de una alerta, dejando fuera sus metadatos.

### 3.2.1. Bases de datos astronómicas

Para validar los experimentos realizados, se usan bases de datos generadas a partir de los estudios astronómicos HiTS y ZTF, ya que estos poseen etiquetas tanto para objetos reales como para bogus. El utilizar dos estudios astronómicos busca aumentar la robustez de los resultados obtenidos y que estos tengan mayores probabilidades de funcionar en estudios astronómicos futuros como ATLAS o LSST. Más detalles del proceso de evaluación de los modelos y por qué usar dos bases de datos pueden ser consultados en la Sección 3.2.6. A continuación se describen brevemente los estudios astronómicos utilizados y las bases de datos correspondientes.

- **HiTS:** Estudio astronómico del año 2013 [26], contempla la observación de 40 campos del cielo en la banda $u$, observaciones realizadas, aproximadamente, cada 2 horas. La base de datos de HiTS contiene 1,604,174 muestras etiquetadas, donde la mitad corresponden a la clase *inlier* (similares a objetos astronómicos reales) y la otra mitad a bogus. Las 802,087 muestras bogus son alertas escogidas al azar, en locaciones donde no se conocían objetos astronómicos al momento de ser observadas, por lo cual se estima que existe un porcentaje de contaminación de *inliers* etiquetados como bogus, sin embargo, se estima que esta contaminación en la clase bogus no excede el 0.2 %. El objetivo de este estudio astronómico era la investigación de SNe, pero como las SNe son objetos astronómicos escasos, y se deseaba generar una base de datos balanceada, los *inliers* fueron simulados al inyectar estrellas de observaciones reales, en imágenes de ciencia aleatorias. Cada muestra consiste de 4 imágenes de 21×21 pixels, imágenes de ciencia, referencia, diferencia y diferencia SNR. En la Figura 3.3 pueden verse ejemplos de cómo lucen las imágenes de muestras de HiTS.

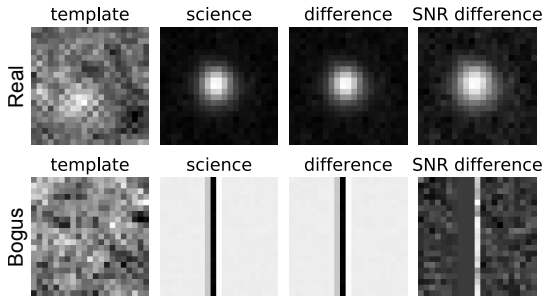



Figura 3.3: Ejemplos de muestras de HiTS. Arriba una muestra de tipo *inlier* y abajo una muestra bogus. Figura obtenida de [4].

**ZTF:** El estudio astronómico ZTF ha sido descrito en la Sección 2.1.1. Sus muestras son generadas por un telescopio que aún se encuentra operativo, con un flujo de $\sim 1$ millón de muestras por noche. A partir de este flujo, ALeRCE ha generado una base

de datos (la que será referida como All-ALeRCE-ZTF en el resto del documento) con muchos objetos y múltiples muestras por objeto, etiquetados en clases obtenidas de hacer *cross-match* con una serie de catálogos astronómicos. Desde All-ALeRCE-ZTF se han generado diferentes bases de datos, cómo ALeRCE-ZTFv5 descrita en la Sección 3.1.1, compuesta por imágenes y metadatos.

Para la detección de anomalías se utiliza las bases de datos ALeRCE-ZTFv1. Las muestras de esta base de datos están compuestas sólo por las 3 imágenes de ciencia, referencia y diferencia, cada una de tamaño 63×63 pixeles. La base de datos ALeRCE-ZTFv1 fue generada en 09/2019 y su distribución de muestras por clase es: 9996 AGN, 9938 VS, 1328 SN, 9899 asteroides y 5350 bogus (1980 bogus de paso 1 y 3370 bogus de paso 2), i.e., se cuenta con 31,161 *inliers*. En esta base de datos los bogus han sido etiquetados de forma visual por expertos, y los *bogus del paso 2* son particularmente parecidos a las muestras reales tipo SN, debido a que fueron etiquetados como SN por versiones iniciales del stamp classifier. En la Figura 3.1 pueden verse imágenes de cada una de las clases de las bases de datos de ZTF, donde las clases AGN, VS, SN y asteroide son consideradas *inliers*.

En el resto del documento cuando se habla de *inliers*, se refiere a objetos astronómicos reales. Mientras que las anomalías astronómicas a detectar son la clase bogus y serán referidas como tal.

### 3.2.2. Partición de datos

Como se mencionó en la Sección 2.3.5, el algoritmo original de GeoTransform utiliza 72 transformaciones geométricas para generar el conjunto de datos auto-etiquetado $S_{\mathcal{T}}$, y su implementación original (disponible en `https://github.com/izikgo/AnomalyDetectionTransformations`) genera todas las transformaciones de una vez, guardándolas en memoria RAM. Esto hace difícil su entrenamientos sobre grandes bases de datos como HiTS o diferentes versiones de ALeRCE-ZTF. Por ejemplo, ALeRCE-ZTFv1 que tan solo posee un total de 36,511 muestras, pesa 2.3 Gb y su versión auto-supervisada con 72 transformaciones pesa 165.45 Gb, cantidad de RAM no disponible en los computadores dispuestos para esta investigación. Se ha propuesto una solución preliminar a este problema técnico, la cual permite entrenar GeoTransform sobre un número arbitrariamente grande de muestras sin embargo, se necesita una investigación más en profundidad para corroborar su correcto funcionamiento, por lo cual ha sido relegada a la Sección B.1 del Apéndice, donde se detalla su funcionamiento y se muestran algunos resultados experimentales preliminares.

Para poder comparar las mejoras propuestas a GeoTransform sobre bases de datos astronómicas sin llenar la RAM de los computadores utilizados, es que se utilizan versiones reducidas de las bases de datos, esto también permite realizar un gran volumen de pruebas diferentes, debido a los reducidos tiempos de entrenamiento. Para poder identificar versiones reducidas de las bases de datos, estas son referidas en el texto agregando el prefijo *small* al nombre de la base de datos.

En la Tabla 3.3 se muestran las particiones en conjuntos de entrenamiento, validación y test utilizadas para cada base de datos.

Tabla 3.3: Particiones de datos utilizadas para las versiones reducidas de las base de datos, estas versiones son reducidas en cantidad de muestras para ser usadas con el esquema original de entrenamiento de GeoTransform.

| Nombre de base de datos | Conjunto de entrenamiento [# de muestras] | Conjunto de validación [# de muestras] | Conjunto de test [# de muestras] |
|---|---|---|---|
| small-HiTS | 7000 | 1000 | 4000 |
| small-ALeRCE-ZTFv1 | 7000 | 1000 | 6000 |

Todos los conjuntos de entrenamiento y validación de la Tabla 3.3 están compuesto únicamente por *inliers* (objetos astronómicos reales), mientras que los conjuntos de test son balanceados con mitad *inliers* y mitad bogus.

Para la base de datos small-ALeRCE-ZTFv1, los *inliers* (SN, AGN, VS y asteroides) son escogidos al azar para formar cada conjunto.

Los bogus en small-ALeRCE-ZTFv1 están compuestos por 1,500 *bogus del paso 1*, y 1,500 *bogus del paso 2*. Los *bogus del paso 2* se consideran difíciles de detectar como anomalías, ya que son más parecidos a los *inliers*, mientras que los *bogus del paso 1* son distinguibles a simple vista de un *inlier*.

### 3.2.3. Pre-procesamiento

El pre-procesamiento indicado a continuación se realiza tanto para las versiones *small* como *large* de las bases de datos HiTS y ALerCE-ZTF.

- **HiTS:** Cada imagen de una muestra de las bases de datos HiTS se normaliza individualmente para tener valores de pixeles en el rango $[-1, 1]$.
- **ALeRCE-ZTF:** Se utiliza el mismo pre-procesamiento aplicado a las imágenes del stamp classifier (Sección 3.1.2), donde se descartan imágenes no cuadradas, se recortan a $21\times21$, se normalizan sin considerar pixeles NaN y se reemplazan los NaNs por 0. La única diferencia es que en este caso se normalizan las imágenes en el rango $[-1, 1]$, siguiendo los lineamientos del trabajo original de GeoTransform [21]. El tamaño de recorte se escoge de $21\times21$ ya que reduce la dimensionalidad del problema y resultados preliminares muestran que logra los mejores resultados para el stamp classifier.

### 3.2.4. Modelo de detección de anomalías GeoTransform

Se utiliza el modelo GeoTransform [21] como punto de partida para detección de bogus en esta tesis, cuyo funcionamiento detallado se encuentra en la Sección 2.3.5. De forma resumida, este algoritmo aplica 72 transformaciones geométricas sobre las bases de datos en las que se entrena y les asigna las etiquetas de las transformaciones aplicadas, luego se entrena un clasificador para discriminar entre las transformaciones aplicadas. Al momento de evaluar una muestra, se le aplican todas las transformaciones y se evalua que tan bien lo hace el clasificador para identificar cada transformacion aplicada, mientras mejor lo hace, recibe un puntaje más alto de normalidad, mientras que si tiene muchas clasificaciones incorrectas

obtiene un bajo puntaje de normalidad, asociándoe a una muestra más anómala.

En esta investigación se le hacen una serie de modificaciones a GeoTransform para mejorar su desempeño. Estas modificaciones se enfocan en cambiar los esquemas de entrenamiento que utiliza (como incluir detención temprana, o *early stopping* en inglés) y estudiar el efecto de seleccionar o incluir transformaciones al conjunto original de 72 transformaciones que utiliza.

### 3.2.5. Arquitectura y esquemas de entrenamiento de GeoTransform

Como se mencionó en la Sección 2.3.5, la arquitectura original de GeoTransform es un clasicador llamado WRN, esta arquitectura es la utilizada en la mayoría de los experimentos, sin embargo en métodos de selección de transformaciones, donde se entrenan una gran cantidad de clasificadores binarios, se necesitan resultados rápidos sobre un problem de baja complejidad por lo que se utiliza como clasificador una CNN con arquitectura similar al clasificador Deep-Hits (DH) [25], pero sin aplicar rotaciones a las imágenes de entrada. En el resto del documento, siempre que se usa DH como clasificador, en vez de WRN, se menciona de forma explícita.

El modelo original de GeoTransform, utiliza un conjunto de $|\mathcal{T}| = 72$ transformaciones geométricas. Como la cantidad de transformaciones utilizadas cambia en diferentes experimentos, el número de transformaciones utilizadas para cada modelo se referencia en su nombre, por ejemplo GeoTransform72. Cada vez que se referencie un modelo nuevo, este será nombrado con el formato GeoTransform*Número_de_transformaciones*.

En lo que respecta al entrenamiento de GeoTransform, para todos los modelos utilizados se entrena con un tamaño de *batch* de 128, optimizador Adam y entropía cruzada como función de pérdida. La cantidad de iteraciones o épocas durante las que se entrena cada modelo varía según el experimento realizado o el objetivo para el que se utilizará el modelo entrenado. En el resto del documento, se referirá a las variaciones en número de iteraciones y/o épocas como el esquema de entrenamiento. A continuación se describen los 2 esquemas de entrenamiento generales a seguir a lo largo de los diferentes experimentos, y en qué casos será utilizado cada uno. Como se mencionó brevemente en la Sección 3.2.2, existe un tercer esquema de entrenamiento llamado *large*, el que permite entrenar GeoTransform sobre bases de datos arbitrariamente grandes, sin embargo esta es una propuesta preliminar y necesita ser explorada en profunidad, detalles y resultados preliminares de esta propuesta pueden verse en la Sección B.1 del Apéndice.

#### 3.2.5.1. Esquema de entrenamiento original

El esquema de entrenamiento original de GeoTransform se detalla en 3.1.6 y consiste en imitar el mismo número de iteraciones que se lograrían si es que el clasificador utilizado fuese entrenado sobre el conjunto de datos original (sin ser aumentado con transformaciones) por 200 épocas. Para GeoTransform72 esto equivale a $200/72 = 2.\bar{7} \approx 3$ épocas. El esquema de entrenamiento original se utiliza exclusivamente sobre GeoTransform72, para ser utilizado como un modelo base con el que comparar las mejoras propuestas. Cuando se utilice este esquema de entrenamiento se referirá al modelo como "original GeoTransform72".

#### 3.2.5.2. Esquema de entrenamiento con detención temprana

En [115] se muestra que en el contexto de aprendizaje auto-supervisado, existe una correlación directa entre la calidad del clasificador y el desempeño en la tarea final a resolver. Entendiéndose en el caso de GeoTransform por "calidad del clasificador" el qué tan bien este identifica las diferentes transformaciones aplicadas, y la tarea a resolver es la obtención de un alto AUROC para el puntaje de normalidad. La calidad del clasificador en GeoTranform no es algo a lo que se preste especial atención, ya que el esquema de entrenamiento original fija un número arbitrario de iteraciones, y nada previene al modelo de sobre-ajustarse al conjunto de entrenamiento.

Obtener un clasificador de calidad y evitar el sobre-ajuste en el contexto de detección de anomalías es una tarea no trivial, ya que al momento de entrenar y validar que tan bien lo está haciendo el clasificador no se tiene acceso a la clase anomalía, lo que impide tener una buena aproximación al desempeño en la tarea final. Esto hace que escoger una estrategia y/o conjunto de validación adecuada sea complejo. En este trabajo se propone utilizar una estrategia de detención temprana al medir el valor de la entropía cruzada para un conjunto de validación compuesto exclusivamente por *inliers*.

Cada vez que se entrena un modelo con detención temprana, se utiliza paciencia (en inglés *patience*) 0, verificando el valor de la entropía cruzada en el conjunto de validación al final de cada época. *Patience* 0 significa que cuando la entropía cruzada en validación aumenta, el entrenamiento es inmediatamente detenido. Se escoge *patience* 0 debido al volumen de experimentos que se realiza, donde se entrena una gran cantidad de modelos y *patience* 0 permite verificar sobre-ajuste de forma temprana y rápida.

### 3.2.6. Evaluación de modelos de detección de anomalías

Siguiendo las definiciones dadas en la Sección 2.3.3, en este trabajo se utiliza el enfoque de detección de anomalías semi-supervisada, donde se cuenta con conjuntos de entrenamiento y validación compuesto exclusivamente por *inliers*, el único momento en que los algoritmos tienen acceso a anomalías es al evaluar el conjunto de test. Debido a esto, al momento de hacer comparaciones entre modelos, se hace difícil escoger el mejor, ya que no se tiene acceso al desempeño real del modelo hasta el final. En este trabajo no se intenta proponer una respuesta a esta problemática por lo cual **en ningún momento se da un veredicto final de cual es el mejor modelo**, sólo se muestran los resultados en los diferentes conjuntos de test y para tareas futuras se escogen una serie de modelos con buenos desempeños en ellas. De forma contraria se corre peligro de sobre-ajustarse al conjunto de test. El utilizar conjuntos de test de dos bases de datos (HiTS y ALeRCE-ZTF) permite tomar una decisión más informada de que modelo presenta el mejor desempeño, siendo aquellos modelos que tengan el mejor desempeño en ambas bases de datos los que se pueden considerar como los mejores con mayor certeza. Para todos los modelos de detección de anomalías presentados, los conjuntos de test utilizados están balanceado en sus etiquetas, donde la mitad de las muestras son bogus y la otra mitad *inliers*. El objetivo final de este trabajo es el aprendizaje de un clasificador binario que entregue como salida un valor de 1 para *inliers* y 0 para bogus. Sin embargo, la mayoría de los modelos, incluidos GeoTransform, entregan un puntaje de normalidad $n_S(x) : \mathcal{X} \to \mathbb{R}$, el que mapea el espacio de todas las posibles muestras $\mathcal{X}$ a un escalar, donde mientras más alto el valor del puntaje de normalidad $n_S(x)$ más parecida a

un *inlier* es la muestra $x$. Transformar el puntaje de normalidad en una clasificación binaria es una tarea no trivial y en continua investigación, por lo cual el foco primario de este trabajo es mejorar GeoTransform para encontrar la mejor función de asignación de puntaje de normalidad $n_S(x)$ en términos del área bajo la curva ROC (AUROC, del inglés *area under the receiver operating characteristic curve*), donde se considera como clase positiva a los *inliers*. Esta métrica será la utilizada para evaluar las diferentes mejoras propuestas a GeoTransform y comparar su desempeño con otros algoritmos de detección de anomalías. Si bien existen otras métricas útiles para evaluar puntajes de normalidad en algoritmos de detección de anomalías, como el área bajo la curva de *precision-recall* (AUPR, del inglés *area under de precision-recall curve*), se escoge utilizar sólo el AUROC ya que resulta ser la más utilizada en la literatura para comparar algoritmos de detección de anomalías [21, 15, 19, 22].

En algunos experimentos se calcula el *accuracy* de GeoTransform en el conjunto de test, al transformar el puntaje de normalidad $n_S \in \mathbb{R}$ en una salida binaria. Esto se logra al definir un umbral $\lambda$ sobre $n_S$. El umbral $\lambda$ es calculado en base al conjunto de validación del modelo, y se fija en un valor que hace que $\lambda$ deje el 97.7 % de los *inliers* de validación en el lado derecho del umbral. Esta regla para seleccionar el valor del umbral $\lambda$ tiene su origen en el intervalo de 2-sigma de una distribución normal, donde si se toma como umbral el valor izquierdo del intervalo 2-sigma, 97.7 % de la masa de la distribución normal queda a la derecha de este umbral. De este modo, a toda muestra $x$ con un puntaje de normalidad $n_S(x)$ mayor a $\lambda$ se le asigna la etiqueta 1 al ser considerada un *inlier*, de lo contrario, se clasifica como bogus con etiqueta 0.

### 3.2.7. Propuesta de nuevas transformaciones para GeoTransform

Una de las hipótesis importantes que se plantean en este trabajo es el hecho de que el conjunto de transformaciones adecuadas depende del problema que se este resolviendo y que se pueden diseñar transformaciones para destacar características utilizando conocimiento a-priori del problema para mejorar el desempeño de modelos entrenados. La detección de anomalías en esta tesis se centra en objetos astronómicos y es sabido que los bogus usualmente aparecen en las imágenes de diferencia de una alerta como bordes de alto contraste, pixeles de valores significativamente diferentes al cielo de fondo. Se postula que agregar transformaciones no geométricas como detección de bordes o borrado de bordes a GeoTransform resulta útil para destacar características presentes únicamente en algunos tipos de bogus. Para codificar este conocimiento a-priori respecto a bogus, es que se utilizan transformaciones basadas en filtros *Laplacianos* y *Gaussianos*, de tal modo de producir efectos de detección de bordes y borrado/difuminación de bordes [116], respectivamente. Las transformaciones basadas en filtros reciben el nombre de *transformaciones con kernel* en el resto del documento. La Figura. 3.4 muestra como las transformaciones con kernel propuestas afectan a una muestra bogus de small-ALeRCE-ZTFv1 (un comportamiento similar se obtiene para los bogus de HiTS). Cuando se aplica un filtro Gaussiano los bordes son difuminados, mientras que al aplicar un filtro Laplaciano los bordes son resaltados, y cuando ambos filtros se aplican de forma consecutiva (aplicando el filtro Gaussiano primero) solo los bordes que no son completamente difuminados por el filtro Gaussiano, son resaltados por el filtro Laplaciano.

Para ambos filtros se utilizan tamaños de kernel de 5×5 pixeles, mientras que para el kernel Gaussiano se usa $\sigma = 1$ y para el kernel Laplaciano se usa $\sigma = 0.5$. Estos parámetros

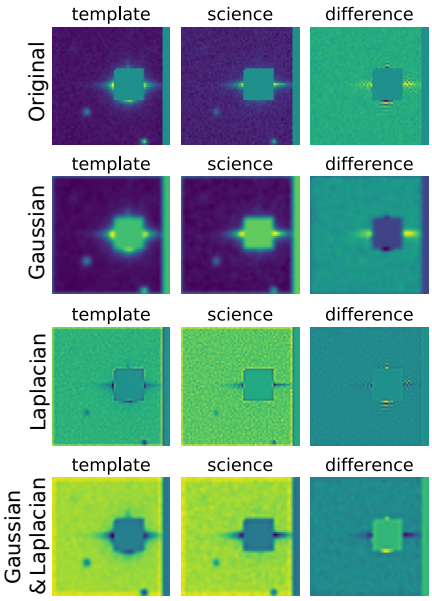



Figura 3.4: Efectos de aplicar transformaciones con kernel basadas en filtros Gaussiano y Laplaciano a un bogus de ALeRCE-ZTFv1. Figura obtenida de [4].

fueron seleccionados de forma visual por un ser humano, de tal forma que se logrará el efecto de borrado y detección de bordes deseado en imágenes con bordes abruptos. Si bien, los parámetros de cada kernel podrían ajustarse de acuerdo a su desempeño para la detección de bogus de GeoTransform, al no contarse con un conjunto de validación con bogus, se prefiere no seleccionar estos parámetros, para no introducir factores extra de varianza al algoritmo que puedan influir en su sobre-ajuste sobre los datos utilizados en el conjunto de test.

### 3.2.8. Selección de transformaciones

En el trabajo original de GeoTransform [21], se utiliza un conjunto de 72 transformaciones geométricas, el cual es escogido de forma arbitraria. Si bien estas transformaciones parecen funcionar bien para bases de datos como CIFAR-10 [63] o Fashion-MNIST [117], este conjunto puede contener transformaciones que no sean realmente necesarias, por lo cual es deseable contar con una metodología para seleccionar el mejor conjunto de transformaciones presentes en la base de datos a utilizar. Con esto se busca probar una de las hipótesis de este trabajo que dice que el mejor conjunto de transformaciones para GeoTransform depende de la base de datos sobre la cual se está trabajando.

Esta investigación se centra en seleccionar las mejores transformaciones para GeoTransform, a partir de un conjunto $\mathcal{T}_\text{i}$ de transformaciones iniciales, el que siempre comprende la transformación identidad $T_0$, i.e., los datos originales. Se aborda el problema de buscar un subconjunto de transformaciones $\mathcal{T}_f \subseteq \mathcal{T}_\text{i}$, a través de descartar aquellas transformaciones que no mejoran o disminuyen su desempeño en la tarea final, se refiere a este tipo de transformaciones como *transformaciones inútiles* $T_{inutil}$. Eliminar transformaciones inutiles tiene el beneficio agregado de usar menos recursos y tiempo en el cálculo de transformaciones extra, logrando que GeoTransform sea más rápido, además de reducir la complejidad de su espacio de clasificación.

A continuación se presentan las motivaciones y definiciones base que impulsan el seleccionar transformaciones para mejorar el desempeño de GeoTransform, y se define un marco empírico con estrategias par la selección de transformaciones.

#### 3.2.8.1. Factores que motivan la selección de transformaciones y definiciones base

En la literatura de aprendizaje auto-supervisado [115], se sostiene que la calidad del clasificador obtenido en el proceso de aprendizaje auto-supervisado, es directamente proporcional al rendimiento de este clasificador en la tarea final que se desea resolver. También se muestra que si se tienen transformaciones redundantes (ya existentes en la base de datos o repetidas en $\mathcal{T}_{\text{i}}$, en el resto del documento se considera a transformaciones redundantes como un tipo de $T_{inutil}$) o que no aportan información útil (transformaciones inútiles $T_{inutil}$), disminuye el desempeño del clasificador en la tarea de predecir las etiquetas generadas para el conjunto auto-supervisado y por ende en la tarea final que se busca resolver, que en el caso de [115] resulta ser transferencia de aprendizaje en clasificación multiclase. Esto demuestra la existencia de un subconjunto de transformaciones que genera mejores resultados $\mathcal{T}_f; \mathcal{T}_f \subseteq \mathcal{T}_{\text{i}}$. En el resto del documento cuando se refiere a etiquetas generadas para el conjunto auto-supervisado, se hablará de etiquetas auto-supervisadas, y a su aprendizaje y estimación como tarea de aprendizaje auto-supervisado.

Ya que GeoTransform es un algoritmo que utiliza aprendizaje auto-supervisado se espera que las premisas de existencia de $\mathcal{T}_f$ le sean aplicables, pese a que la tarea final de GeoTransform es aumentar el AUROC en detección de anomalías, a diferencia de lo presentado en [115]. Teniendo en cuenta, que en [115] se identifican las transformaciones redundantes y las que no aportan información como factores que afectan la calidad de un clasificador $f_\theta$ de parámetros $\theta$, es que a continuación se definen estos conceptos como factores de calidad.

**Definición 3.2.1** (Factores de calidad)**.** Los factores que afectan la calidad de un clasificador $f_\theta$ de parámetros $\theta$, en el contexto de aprendizaje auto-supervisado son dos, y quedan definidos por

- **Factor de calidad 1**: Que las transformaciones usadas contengan información útil para aumentar el desempeño en la tarea final para la que se usará el clasificador entrenado de forma auto-supervisada.
- **Factor de calidad 2**: Que las transformaciones utilizadas no sean redundantes.

El factor de calidad 1 se refiere a qué tan útiles resultan las transformaciones para codificar información presente en los datos originales (*inliers*), que permitan al clasificador generar representaciones para mejorar el desempeño en la tarea final (detección de anomalías), y que su presencia no disminuya el desempeño del algoritmo en la tarea final. La existencia de este factor es difícil de probar, ya que medir qué tan útiles son las transformaciones, sin evaluarlas en la tarea final, es un problema no trivial, por lo cual este factor sólo será explorado en términos de medir el efecto de transformaciones que borran totalmente la información presente en las muestras.

El factor de calidad 2 se refiere a que las transformaciones no estén previamente contenidas en el conjunto de datos original, ni que en el conjunto de transformaciones utilizado $\mathcal{T}_{\text{i}}$ existan

transformaciones que generen efecto similares. Este factor para generar un clasificador de calidad $f_\theta$ tiene que ver con la hipótesis *Visual Transformation Self-Supervision (VTSS)* [115], la que se define textualmente como:

*Dado un conjunto de datos sobre el que se realiza una tarea de aprendizaje auto-supervisado, mientras se predigan instancias de una transformación, si es que las instancias parecidas de las transformaciones ya están presentes en la base de datos, entonces la representación será menos útil.*

En [115], se muestra que existe una proporcionalidad indirecta entre la cantidad de transformaciones previamente contenidas en los datos originales y el desempeño en la tarea final (*accuracy* semi-supervisado).

Al utilizar transformaciones ya existentes en la base de datos, se tiene que el clasificador será incapaz de distinguir entre una instancia transformada y otra no transformada, obteniendo un bajo *accuracy* para estos casos, y lo que es peor, se le forzará a aprender una etiqueta que le es imposible ajustar de forma adecuada, la que oscilará entre la instancia transformada y no transformada, lo que a su vez conlleva a un aprendizaje de filtros deteriorados por un proceso de aprendizaje ruidoso.

A continuación se profundiza en el concepto de invarianza, el que permite detectar transformaciones redundantes, ligadas al factor de calidad 2.

#### 3.2.8.2. Definiciones de invarianza

La existencia previa de transformaciones en un conjunto de datos $\mathcal{X}$ (en el resto del documento se referirá a los conjuntos de datos como $\mathcal{X}$, los que están compuestos por muestras $x$, que a su vez están compuestas por imágenes), no necesariamente tiene que estar ligada a la existencia explícita de versiones transformadas de los datos en $\mathcal{X}$. Sino que también puede estar relacionada a propiedades mismas de las estructuras de los datos, como lo es la invarianza perfecta, concepto que se define como:

**Definición 3.2.2** (Invarianza perfecta)**.** Una imagen $x$ se dice perfectamente invariante a una transformación $T$ si es que se cumple

$$x = T(x). \tag{3.2}$$

Si es que un conjunto de datos $\mathcal{X}$ es invariante a una transformación $T$, cada imagen de $\mathcal{X}$ será exactamente igual tras la transformación, y puede considerarse que $T$ existía de forma previa en $\mathcal{X}$.

En conjuntos de datos del mundo real, la existencia previa de transformaciones exactas es un fenómeno poco común. Normalmente, si pre-existen transformaciones, por ejemplo, variaciones de ángulos en las imágenes, estas contienen factores de variación que hacen que no sean una simple rotación de una imagen ya presente en $\mathcal{X}$, además no siempre todas las imágenes de $\mathcal{X}$ serán sometidas a las mismas transformaciones pre-existentes. Esto también ocurre para la invarianza de imágenes a ciertas transformaciones, donde es raro encontrar

bases de datos perfectamente invariantes a alguna transformación, normalmente existen leves diferencias. Cuando las diferencias entre una muestra $x$ y su versión transformada $T(x)$ son suficientemente pequeñas, de tal modo que a simple vista estas son indistinguibles la una de la otra, se habla de invarianza imperfecta, definida por:

**Definición 3.2.3** (Invarianza imperfecta)**.** Una muestra $x$ se dice imperfectamente invariante a una transformación $T$ si es que se cumple

$$x \approx T(x). \tag{3.3}$$

En el resto del documento se referirá a invarianza imperfecta, simplemente como invarianza.

En lo anterior se muestra la definición utilizada en esta investigación para el concepto de invarianza para una muestra. A continuación se presenta la definición utilizada de que un conjunto de datos $\mathcal{X}$ sea invariante a una transformación $T$.

**Definición 3.2.4** (Invarianza de un conjunto de datos)**.** Un conjunto de datos $\mathcal{X}$ se dice invariante (imperfectamente invariante) a una transformación $T$ si es que se cumple $\mathcal{X} \approx T(\mathcal{X})$, donde $T(\mathcal{X})$ es un conjunto de datos en el que a cada muestra $x \in \mathcal{X}$ se le ha aplicado $T$. La medida de similitud $\approx$ depende del método que se utilice para medir similitud, el cual puede ser medido al comparar cada muestra $x$ con su versión transformada o una medición sobre todas las muestras y sus versiones transformadas.

Un ejemplo de conjuntos de datos que *teóricamente* presentan invarianzas a transformaciones geométricas son las imágenes de objetos astronómicos reales, ya que, por su naturaleza y ubicación en un espacio sin un punto de referencia fijo, la identificación del tipo de objeto que es observado es independiente de la orientación desde donde se capture su imagen, además de que la mayoría de los objetos astronómicos aparecen como objetos puntuales, especialmente cuando estos están muy lejos o los telescopios utilizados para tomar las imágenes son de baja resolución. La base de datos HiTS contiene objetos, que como puede apreciarse en la Figura 3.3 de la Sección 3.2.1, son puntuales en el centro de las imágenes para la clase *inlier* (objeto real), por lo cual puede considerarse que contiene invarianza tanto a rotaciones como a *flips* (volteos de una imagen en algún eje que pasa por el centro de ella) en cualquier eje de las imágenes. Por lo que para la clase *inlier* se dice que se tiene invarianza en el conjunto de datos.

En la práctica, pueden existir cierto tipo de objetos astronómicos donde orientaciones particulares entregan más información para facilitar su identificación. Por ejemplo, es más fácil diferenciar si una galaxia es espiral o elíptica si es que es observada de forma perpendicular a su zenit, que si se observa a lo largo de sus ejes galácticos, caso en que ambos tipos de galaxias se observan como un trazo lineal. La existencia de estos sesgos en orientación se puede transmitir a las bases de datos generadas, particularmente a las de imágenes con alta resolución, donde se pueden distinguir los objetos astronómicos y estructuras adyacentes como mucho más que objetos puntuales. De este modo es que una base de datos astronómica puede perder o disminuir su invarianza rotacional o a *flips*. Un ejemplo de esto es la base de datos ALeRCE-ZTF, donde en la Figura 3.1 se aprecia que objetos reales como SNe tienden

a presentar galaxias adyacentes y sólo por un sesgo en la base de datos presente, la mayoría de las SNe presentan su galaxia a la derecha, disminuyendo así esta base de datos su invarianza frente a *flips* en el eje vertical o a rotaciones para la clase SN, pese a que una pequeña cantidad de las muestras puedan presentar este sesgo, la base de datos en su totalidad no es completamente invariante.

#### 3.2.8.3. Selección de transformaciones desde la teoría

Al considerar el factor de calidad 2 para un clasificador entrenado bajo auto-supervisión, es tentador buscar transformaciones previamente contenidas en los datos y descartarlas como redundantes, particularmente aquellas a las que los datos son invariantes. Sin embargo, como se explica en la Sección 3.2.8.2, la invarianza perfecta es una cualidad que raramente existe en bases de datos reales, normalmente solo se obtiene invarianza imperfecta. Por esto, antes de descartar una transformación a las que los datos son invariantes o parcialmente invariantes, es necesario evaluar la sinergia existente entre transformaciones a las que el conjunto de datos es invariante y el resto de las transformaciones. La sinergia se tiene cuando dos variables aleatorias que contienen diferente información, al ser combinadas tienen una información que es mayor que la suma de las partes. Con esto, transformaciones que por si solas se consideran redundante por invarianza, pueden presentar sinergias con otras transformaciones, las que ayuden a identificar ciertos factores de variación en algunas muestras, mejorando el desempeño en la tarea final de la estrategia auto-supervisada que es la detección de bogus.

Un ejemplo de la sinergia que pueden aportar ciertas transformaciones a las que una base de datos resulta invariante, es el caso del clasificador real-bogus Deep-Hits (DH) [25], donde las imágenes que entran al clasificador son rotadas en 4 orientaciones (múltiplos de 90°) para ser procesadas al mismo tiempo por el clasificador, lo que aumenta con creces su desempeño sobre la base de datos HiTS. En este caso, pese a que existe invarianza a las rotaciones en HiTS, no es invarianza perfecta, por lo cual utilizar versiones rotadas de las imágenes otorga información extra al modelo, lo cual le da robustez y aumenta su capacidad de generalización, aumentando su desempeño. Si es que una base de datos presenta invarianza perfecta a cierta transformación, incluirla sería redundante y podría ser descartada, pero una transformación con invarianza imperfecta no puede ser descartada así como así, ya que su sinergia con datos normales puede ser útil para la tarea que se está resolviendo.

Para evidenciar los efectos de sinergias, se hace fundamental analizar el efecto de la cantidad de información para la detección de anomalías y el análisis de estrategias auto-supervisadas desde el punto de vista de la teoría de información [118] y la literatura relacionada a selección de características [119]. Debido al tiempo acotado de esta investigación se plantea como trabajo futuro desarrollar un cimiento teórico para evaluar sinergias entre transformaciones y determinar cuando realmente una transformación es relevante para la tarea de detección de anomalías.

Debido a esto, es que se ataca la búsqueda de transformaciones relevantes desde un punto de vista puramente empírico, al proponer métodos que sirvan para seleccionar transformaciones que funcionen en la práctica y hagan sentido con los factores de calidad de la Definición 3.2.1.

En el resto del documento se obvian las sinergias entre transformaciones y sólo se tra-

baja desde métodos empíricos que mejoran el desempeño del modelo en su tarea final, sin preocuparse si es que las sinergias otorgan mejores alternativas.

#### 3.2.8.4. Selección empírica de transformaciones

La selección empírica de transformaciones busca encontrar soluciones que satisfagan y prueben los factores 1 y 2 de calidad de un clasificador (Definición 3.2.1), los cuales se han de cumplir en el conjunto de transformaciones $\mathcal{T}$ para mejorar la calidad de clasificador $f_\theta$.

Se trabaja desde la selección o descarte de transformaciones a partir de un conjunto de transformaciones inicial $\mathcal{T}_\text{i}$. Se da particular énfasis a la búsqueda y descarte de transformaciones a la que cierta base de datos resulta invariante, y por simplicidad, obviando un análisis de sinergias, el cual se sabe que es necesario en un trabajo futuro. Se considera que un método es útil por sus efectos sobre el desempeño de GeoTransform en el AUROC de test. Se ha de tener en cuenta que estos métodos pueden ser catalogados como útiles por efectos de sobre-ajuste, sin embargo, debido a la problemática de validar detección de anomalías sin el conjunto de test, no se espera solucionar esta problemática y se deja como trabajo futuro.

A continuación se presenta la definición de transformaciones inútiles propuesta en este trabajo, donde para cada tipo de transformaciones inútiles se presentará un método empírico para su descarte. Se ha de tener en consideración que estas transformaciones se consideran inútiles basándose en los factores de calidad de un clasificador y sólo tienen validez a nivel empírico.

**3.2.8.4.1. Definición del problema y de transformaciones inútiles:** El problema de selección empírica de transformaciones se reduce a:

Dado un conjunto de transformaciones inicial $\mathcal{T}_\text{i}$ a ser aplicadas sobre un conjunto de datos $\mathcal{X}$ para entrenar un clasificador $f$ de forma auto-supervisada, encontrar métodos $M_\text{i}$ para identificar transformaciones *inútiles* $T_{\text{inutil}}$ y descartarlas de $\mathcal{T}_\text{i}$, para llegar a un $\mathcal{T}_f$ que satisfaga los factores de calidad de $f$, aumentando su desempeño en la tarea final de AUROC en test.

**Definición 3.2.5** (Transformación inútil $T_{\text{inutil}}$)**.** Una transformación $T$ es categorizada como una *transformación inútil* $T_{\text{inutil}}$ si es que al ser aplicada sobre todas las muestras de $\mathcal{X}$, esta cumple uno o más de los siguientes criterios:

1. $T_{\text{inutil}.1}$: Borra toda la información contenida en el conjunto de datos original $\mathcal{X}$.
2. $T_{\text{inutil}.2}$: Transformación redundante que ya está presente en el conjunto de datos original $\mathcal{X}$ o existe otra transformación en $\mathcal{T}$ que produce un efecto similar, i.e. $\exists T \in \mathcal{T};\, T(\mathcal{X}) \approx T_{\text{inutil}.2}(\mathcal{X})$.
3. $T_{\text{inutil}.3}$: Cuando se agrega al conjunto de transformaciones $\mathcal{T}_\text{i}$, baja el desempeño del modelo $f$, en la tarea final que este busca resolver.

Los criterios 1 y 3 están relacionados al factor de calidad 1 donde se buscan transformaciones de calidad que contengan información útil. Para el criterio 1 $T_{\text{inutil}.1}$ se propone un método basado en teoría de la información para medir si se ha borrado la información. El criterio 2 $T_{\text{inutil}.2}$ se relaciona al factor de calidad 2, buscando evitar transformaciones inútiles

y redundantes que producen un entrenamiento errático donde se aprenden filtros deteriorados o ruidosos, donda la redundancia puede provenir desde invarianzas en el conjunto de datos. Para este criterio $T_{inutil.2}$ se proponen métodos que midan similitud entre transformaciones. Para encontrar $T_{inutil.3}$ se proponen métodos que miden el efecto de remover o mantener alguna transformación.

A continuación, se definen métodos para encontrar los distintos tipos de transformaciones inútiles, lo cuales son propuestos para satisfacer cada definición de transformación inútil y son validados de forma empírica. Cada método tiene un relación directa con el tipo de transformación inútil que busca encontrar y se basan en evaluar de forma individual cada transformación $T_j \in \mathcal{T}_{\text{i}}$. Se ha de recordar que la transformación identidad $T_0$ nunca se descarta como inútil.

Para validar el éxito o fracaso de los métodos presentados a continuación se utiliza un conjunto inicial de transformaciones $T_{\text{i}}$ donde se tienen transformaciones a las que se sabe que los *inliers* de HiTS son invariantes, evaluándose la capacidad de cada método para eliminar las transformaciones para las que fue diseñado eliminar. Detalles de esta estrategia para validar los métodos se muestran en la Sección 4.2.3 de resultados. Además, se considera como útil un método, si es que este aumenta el desempeño en el conjunto de test de las 2 bases de datos utilizadas en cada caso, sin declararlo un éxito ya que la mejora en desempeño podría ser un efecto de sobre-ajuste.

**3.2.8.4.2. Método de selección 1 - Información mutua:** El método $M_1$ se diseña para descartar transformaciones inútiles del tipo $T_{inutil.1}$, i.e., transformaciones que borran toda la información original de una muestra. Ya que se buscan medidas relacionadas a la información entre dos variables es que se utiliza la información mutua (MI del inglés *mutual information*) para diseñar $M_1$. Debido a que la MI se mide entre 2 variables aleatorias, en vez de medir la MI entre una muestra $x$ y su versión transformada $T(x)$, se mide la MI entre toda el conjunto de datos original $\mathcal{X}$ y su versión transformada, donde si $\text{MI}(\mathcal{X}; T(\mathcal{X})) \sim 0 \implies T \equiv T_{inutil.1}$. Debido a que la MI es una magnitud que no puede ser directamente calculada sobre distribuciones de datos desconocidas y de alta dimensionalidad, es que se estima utilizando una versión modificada del estimador llamado *matrix-based Renyi's entropy* [120], originalmente propuesto en [121]. Este método utiliza un kernel para estimar la MI y tiene un orden computacional de $O(n^2)$. El valor final que se reporta al utilizar este estimador de MI es el promedio al ser calculado sobre *batches* de tamaño 512. En el resto del documento, siempre que se refiera al cálculo de MI será sinónimo de utilizar la metodología descrita. Utilizando este estimador de MI, finalmente se define $M_1$ como:

**Definición 3.2.6** (Método de selección $M_1$)**.** Sea $T$ una transformación aplicada sobre todas las muestras de $\mathcal{X}$, $M_1$ estipula:

$$\frac{MI(\mathcal{X}; T(\mathcal{X}))}{H(\mathcal{X})} < \varepsilon \implies T \equiv T_{inutil.1}, \tag{3.4}$$

donde $\text{H}(\mathcal{X})$ corresponde a la entropía de $\mathcal{X}$ y $\varepsilon = 10^{-2}$ es un umbral determinado de forma experimental.

La Definición 3.2.6 incluye la entropía $H(\mathcal{X})$ en el denominador ya que este término equivale a calcular la "auto-MI" $\text{MI}(\mathcal{X}; \mathcal{X})$, y se usa para señalar que si $T \equiv T_{inutil.1}$, entonces

se debería cumplir $MI(\mathcal{X}; T(\mathcal{X})) \lll H(\mathcal{X}) = MI(\mathcal{X}; \mathcal{X})$, y $\varepsilon = 10^{-2}$ se escoge como el valor de MI entre imágenes de la base de datos CIFAR-10 y las mismas imágenes multiplicadas por 0, donde se sabe que se borra toda la información, y se asume como un buen indicador para funcionar en otras bases de datos. La selección de $\varepsilon$ requiere un análisis en mayor profundidad.

**3.2.8.4.3. Método de selección 2.1 - Invarianza a una transformación:** La identificación de transformaciones inútiles del tipo $T_{inutil.2}$, i.e., transformaciones redundantes, es un problema abierto que se trata en primera instancia en [115], y está ligado al factor de calidad 2 de un clasificador. Debido a la dificultad de encontrar transformaciones arbitrarias ya presentes en la base de datos, y dado que en esta tesis se trabaja sobre datos astronómicos, los cuales se sabe presentan cierto grado de invarianza a rotaciones y *flip*, es que se acota el problema a encontrar transformaciones a las que los datos son invariantes, que como se discute anteriormente en la Sección 3.2.8.2, son un tipo de transformaciones contenidas en los datos. Se refiere a este tipo de transformaciones como $T_{inutil.2.1}$, definidas por:

**Definición 3.2.7** (Transformación inútil tipo $T_{inutil.2.1}$)**.** Una transformación $T$ es categorizada como $T_{inutil.2.1}$ si se cumple alguno de los dos siguientes puntos:

1. El conjunto de datos $\mathcal{X}$ es invariante a $T$, i.e. $\mathcal{X} \approx T(\mathcal{X})$.
2. En el conjunto de transformaciones $\mathcal{T}$ ya existe otra transformación $T_k$ que produce un efecto similar a $T$, i.e. $\exists T_k \in \mathcal{T};\ T_k(\mathcal{X}) \approx T(\mathcal{X})$.

Se ha de recordar que cuando se habla de invarianza, se refiere a invarianza parcial, por lo cual no son transformaciones totalmente redundantes y, al no considerar sinergia con las transformaciones existentes, puede que se elimine información que deteriore el desempeño de la tarea final, en este caso detección de anomalías.

Cuando se buscan métodos para encontrar transformaciones del tipo $T_{inutil.2.1}$, se hace fundamental definir qué tipo de similaridad se busca medir entre el conjunto de datos original $\mathcal{X}$ y sus versiones transformadas $T(\mathcal{X})$, buscando una similaridad a nivel de conjunto completo de datos, tal y como se define para el concepto de invarianza de un conjunto de datos (Definición 3.2.4). De la Definición 3.2.4 el término $\approx$ indica que $\mathcal{X}$ ha de ser *similar* a su versión transformada $T(\mathcal{X})$, pero, ¿Qué significa similaridad?. Como puede verse en la Figura 3.5, existen muchos tipos de similaridad, y no todos son útiles para el problema que se busca resolver.

En la Figura 3.5 se observan diferentes tipos de similaridad para pares de imágenes, donde se aprecia la similaridad de clases, de versiones transformadas de una misma imagen, y de apariencia, siendo esta última la que se busca en este trabajo. Esta similaridad en apariencia también resulta ser ambigua, ya que no esta definida. Además, está la dificultad extra que se busca similaridad a nivel de conjuntos de datos y no sólo imágenes individuales.

Se propone encontrar similaridad de apariencia entre dos conjuntos de datos con un método basado en qué tanto se confunde un clasificador entre ambos conjuntos. Este método busca descartar transformaciones a la que los datos resultan invariantes y comparar pares de transformaciones para evidenciar cuales son redundantes al generan efectos similares.


Figura 3.5: Diferentes tipos de similaridad para pares de imágenes. A la izquierda se tiene similaridad de clases con dos imágenes que contienen perros. Al centro similaridad de contenido con dos imágenes idénticas que difieren en orientación. A la derecha similaridad de apariencia con dos imágenes de ciencia de objetos astronómicos reales diferentes de la base de datos HiTS, donde ambas imágenes lucen idénticas. Este último tipo de similaridad es el que se busca en este trabajo.

**3.2.8.4.4. Método de selección 2.1.a - Matriz de discriminación:** El método $M_{2.1.a}$ busca transformaciones a las que el conjunto de datos original resulta invariante o transformaciones que ya existen en el conjunto inicial de transformaciones. Mide similitud entre transformaciones, donde el conjunto de datos original puede ser considerado como que se le ha aplicado la transformación $T_0$, equivalente a la identidad. $M_{2.1.a}$ mide similitud en base a la capacidad de discriminación de un clasificador $f$ entrenado para diferenciar entre un par de transformaciones a comparar, quedando definido por:

**Definición 3.2.8** (Método de selección $M_{2.1.a}$)**.** Al comparar dos transformaciones $T_\mathrm{i}$ y $T_j$, aplicadas sobre las muestras de $\mathcal{X}$, si es que al entrenar un clasificador para distinguir entre $T_\mathrm{i}(\mathcal{X})$ y $T_j(\mathcal{X})$, y luego evaluarlo sobre un conjunto de validación balanceado con muestras de $T_\mathrm{i}(\mathcal{X})$ y $T_j(\mathcal{X})$, este obtiene un *accuracy* entre 49 % y 51 % (∼50 %), las transformaciones son equivalentes y una resulta redundante, i.e., se tiene una $T_{inutil.2.1}$.

Como caso particular, si es que una de las transformaciones es la identidad $T_0$, entonces $\mathcal{X}$ es invariante a la otra transformación.

Para aplicar $M_{2.1.a}$ se toma el conjunto de datos de entrenamiento auto-etiquetado $\mathcal{X}_\mathcal{T}$ con $|\mathcal{T}|$ transformaciones iniciales, luego se divide en subconjuntos binarios de datos $\mathcal{X}_{\mathcal{T}_{ij}}$, compuestos por pares de transformaciones $T_\mathrm{i}$ y $T_j$, donde i $> j$. Así se obtienen $\frac{|\mathcal{T}|\times(|\mathcal{T}|-1)}{2}$ subconjuntos. Luego, por cada subconjunto $\mathcal{X}_{\mathcal{T}_{ij}}$ se entrena un clasificador con arquitectura Deep-HiTS [25], pero sin utilizar rotaciones de las imágenes a la entrada.

Si el conjunto de datos original $\mathcal{X}$ resulta invariante a alguna transformación, o existen 2 transformaciones que generen un efecto similar sobre los datos, el clasificador $f$ no será capaz de distinguir una transformación de la otra y el *accuracy* resultante es ∼50 %. Al computar el *accuracy* para todos los pares de transformaciones, estos pueden ser visualizados en una *matriz de discriminación* (similar a una matriz de correlación), la que indica si un par específico de transformaciones pueden ser discriminadas o no. Es necesario mencionar que los *accuracies* de la matriz de confusión se calculan sobre conjuntos de validación balanceados, diferentes a los conjuntos de entrenamiento.

Tras computar la matriz de discriminación, para cada par de transformaciones que obten-

gan *accuracy* entre 49 % y 51 %, se selecciona la transformación con el mínimo número de operaciones, descartando la otra como una transformación inútil de tipo $T_{inutil.2.1}$. El razonamiento detrás de esta estrategia y todas las que descartan transformaciones redundantes de a pares, es que no importa realmente cual transformación sea escogida del par, ya que el clasificador no puede distinguir entre ellas.

Para entrenar los clasificadores de pares de transformaciones se utiliza el esquema de entrenamiento con detención temprana, descrito en la Sección 3.2.5.2.

El método $M_{2.1.a}$, recibe este nombre y no simplemente es llamado $M_{2.1}$, ya que durante el desarrollo de este trabajo se exploró otro método de selección similar, llamado $M_{2.1.b}$ y que puede ser consultado en la Sección B.2 del Apéndice. Este último difiere de $M_{2.1.a}$, únicamente en el mecanismo de medición de similitud entre un par de transformaciones, el cual se basa en la distancia entre la distribución de datos presentes entre 2 pares de transformaciones diferentes. Esta distancia se estima calculando la distancia de Frèchet (DF) [122] sobre las muestras del par de transformaciones comparadas. Pese a que se realizaron una serie de experimentos utilizando el método $M_{2.1.b}$, se encontraron ciertas inestabilidades en su funcionamiento, principalmente relacionadas con el ajuste de umbrales. Por estas razones y la necesidad de realizar un estudio en profundidad para el correcto funcionamiento del método $M_{2.1.b}$, es que se relega a la Sección B.2 del Apéndice, donde se define este método en profundidad y se muestran resultados experimentales preliminares.

**3.2.8.4.5. Método de selección 3 - Desempeño sobre la tarea final:** El método 3 para seleccionar transformaciones se basa en una estrategia puramente empírica de prueba y error. Se prueban una a una las transformaciones a utilizar en la tarea auto-supervisada, y si estas bajan el desempeño del algoritmo en la tarea final, la transformación se deja fuera y se categoriza como inútil del tipo $T_{inutil.3}$. Este criterio busca el subconjunto de transformaciones que genere el mayor desempeño en la tarea final.

La principal dificultad de este método es no poder evaluar el desempeño en la tarea final, es decir, AUROC en detección de bogus, ya que no se tiene acceso a los bogus durante el entrenamiento. Para aproximar los bogus en el dominio de datos sobre el que se estén seleccionando transformaciones, se utilizan bogus de otras bases de datos astronómicas, y con esto se mide el AUROC. Si es que se está seleccionando transformaciones para HiTS se usan bogus de ALeRCE-ZTF y si se está seleccionando para ALeRCE-ZTF, se utilizan bogus de HiTS para medir el AUROC. Cada vez que se refiera a medir el AUROC donde los bogus provengan de una base de datos astronómica diferente a la original, se hablará de AUROC-proxy. De este modo se puede entrenar el clasificador auto-supervisado y medir el efecto de las diferentes transformaciones en la tarea final.

La estrategia ideal para encontrar el subconjunto de transformaciones $\mathcal{T}_f \subseteq \mathcal{T}_\text{i}$ que obtenga el mejor AUROC-proxy, sería evaluar todos los subconjuntos de transformaciones posibles, sin embargo abordar el problema de esta forma presenta una complejidad algorítmica de orden $O(2^n)$, ya que se requiere evaluar el algoritmo $\sum_{k=1}^{n} \binom{n}{k} = 2^n - 1$ veces al contar con $n$ transformaciones, sumado al hecho de que cada vez que se evalua el algoritmo, para obtener una medida de desempeño representativa, se usa el AUROC-proxy promedio de 10 entrenamientos.

Con el objetivo de disminuir la complejidad del problema, es que se proponen dos enfoques inspirados en la literatura de selección de características [119]. Estas estrategias responden a enfoque iterativos donde se van probando transformaciones una a una. La primera de ellas es la selección *progresiva*, donde desde el conjunto inicial de transformaciones a evaluar $\mathcal{T}_{\text{i}}$ se extrae la transformación identidad $T_0$ y se agrega a un conjunto $\mathcal{H}$, el cual va a contener las transformaciones finalmente seleccionadas. Luego, se selecciona una transformación $T_j$ y se agrega a $\mathcal{H}$, para entrenar GeoTransform con $\mathcal{H}$ y medir el AUROC-proxy promedio, este proceso se repite para todas las transformaciones $T_j \in \mathcal{T}_{\text{i}} - \mathcal{H}$ que quedan en el conjunto original de transformaciones. Finalmente, se selecciona y se agrega a $\mathcal{H}$ la transformación que alcance el AUROC-proxy más alto. Este proceso se repite para todas las transformaciones en el conjunto inicial $\mathcal{T}_{\text{i}}$, y se usa como conjunto de transformaciones seleccionados $\mathcal{T}_f$ aquel $\mathcal{H}$ que haya obtenido el AUROC-proxy más alto en el proceso completo. Las transformaciones que quedan fuera durante este proceso se denotan como transformaciones inútiles del tipo $T_{inutil.3}$. Esta estrategia cuenta con complejidad algorítmica $O(n^2)$ y recibe el nombre de $M_3 - progresivo$.

Otra estrategia probada, es la selección *regresiva*, que difiere con la estrategia anterior, en que el conjunto $\mathcal{H}$, contiene todas las transformaciones de $\mathcal{T}_{\text{i}}$, i.e., $\mathcal{H} \equiv \mathcal{T}_{\text{i}}$. Además, se van removiendo transformaciones de una en una, evaluando GeoTransform de tal forma que se deja fuera la transformación con peor AUROC-proxy en cada proceso de descartar una transformación. Este proceso se repite hasta que $\mathcal{H}$ queda reducido a la transformación identidad, siendo la única transformación que no esta sujeta a remoción. Finalmente, se selecciona como mejor grupo de transformaciones $\mathcal{T}_f$ aquel $\mathcal{H}$ que obtuvo el AUROC-proxy promedio más alto en el proceso. Aquellas transformaciones que quedan fuera son categorizadas como inútiles de tipo $T_{inutil.3}$. Al igual que la estrategia progresiva, la estrategia regresiva también presenta una complejidad algorítmica $O(n^2)$, sin embargo, como se comienza con todo el conjunto de transformaciones iniciales al comienzo de la selección, esta toma más tiempo en recorrer todos los subconjuntos de transformaciones a evaluar, netamente por que más transformaciones implican un mayor tiempo de entrenamiento para el algoritmo. Este método recibe el nombre de $M_3 - regresivo$.

Debido a que tanto $M_3 - progresivo$ como $M_3 - regresivo$ tienen complejidad algorítmica $O(n^2)$ para la selección de transformaciones, es que estos métodos han de ser evaluados sobre conjuntos de transformaciones iniciales con un número reducido de transformaciones. Además, dado que la utilización de un AUROC-proxy puede no reflejar bien el desempeño del AUROC final, estos métodos se exploran como propuestas preliminares.

**3.2.8.4.6. Pipelines de selección - Mezcla de métodos:** Los métodos presentados no son excluyentes para la selección de transformaciones, por lo cual se propone su agrupación en *flujos de detección*. Estos flujos son utilizados para evaluar los métodos a nivel individual y además comparar qué combinación de métodos logra generar el subconjunto de transformaciones que permite a GeoTransform tener el AUROC más alto en bases de datos astronómicas como HiTS y ALeRCE-ZTF.

Cada vez que se define un flujo donde convive más de un método, estos son aplicados en forma secuencial partiendo desde el método de menor número hasta el de mayor número. Esto se ha concebido de esta manera debido a que los métodos de menor número presentan

tiempos de ejecución menores y tienen un comportamiento más confiable, permitiendo eliminar transformaciones de forma temprana y disminuir el tiempo de cómputo de métodos posteriores. Un ejemplo de posibles flujos de selección que se pueden definir se aprecia en la Figura 3.6.

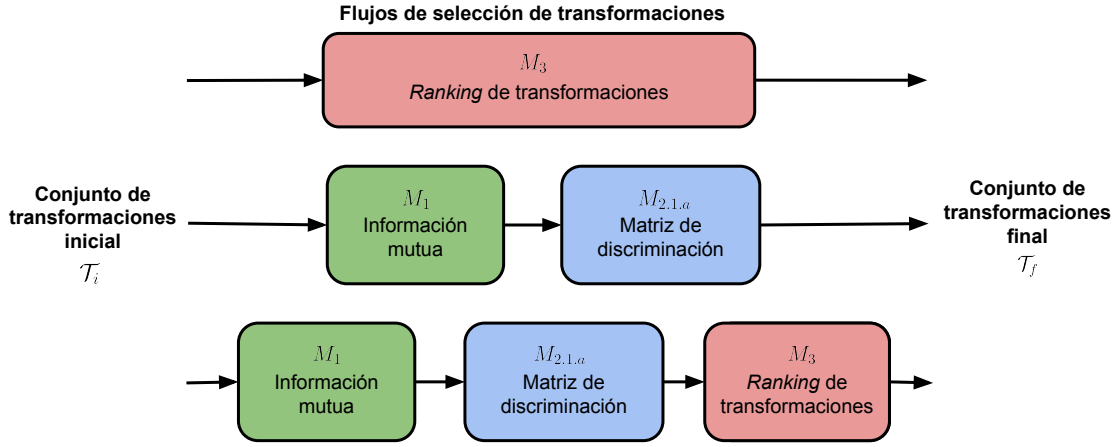



Figura 3.6: Ejemplos de mezclas de métodos en flujos de selección de transformaciones.

### 3.2.9. Comparación entre diferentes detectores de anomalías

Se compara el desempeño de las mejoras propuestas a GeoTransform con diferentes algoritmos de detección de anomalías, usando tanto modelos clásicos como del estado del arte. El desempeño se mide en términos de AUROC promedio de 10 instancias de entrenamiento, sobre las bases de datos small-HiTS y small-ALeRCE-ZTFv1.

Se utilizan 6 modelos diferentes a GeoTransform: OC-SVM, IF, CAE-OC-SVM, ADGAN y MO-GAAL, los que han sido descritos en la Sección 2.3.4. Para los algoritmos de aprendizaje de máquinas clásico como OC-SVM e IF, las imágenes de entrada se reestructuran como vectores planos. Además, debido a que estos algoritmos son sensibles a los hiperparámetros utilizados, se les da una ventaja y los hiperparámetros críticos para cada uno se escogen como los que obtienen el mejor AUROC sobre el conjunto de test en una búsqueda en grilla (se les da la ventaja de mostrar resultados sobre-ajustados al conjunto de test). Para OC-SVM y CAE-OC-SVM, la búsqueda se realiza sobre los hiperparámetros $\gamma$ y $\nu$ con una grilla sobre $\gamma \in \{2^{-7}, 2^{-6}, ..., 2^2\}$, $\nu \in \{0.1, 0.2, ..., 0.9\}$, mientras que para IF se utiliza un búsqueda en la grilla sobre $n\_estimadores \in \{100, 200, ..., 800\}$, $contaminación \in \{0.1, 0.2, ..., 0.5\}$.

## 3.3. Detección automática de bogus para la clasificación de objetos astronómicos

Tras trabajar con el detector de anomalías GeoTransform y proponer mejoras que incrementan su desempeño en bases de datos astronómicas como small-HiTS y small-ALeRCE-ZTFv1, se evalúa su utilidad para detectar bogus de forma automática y mejorar el desempeño de clasificadores supervisados como el stamp classifier, buscando aliviar la tarea de astrónomos en el etiquetado manual de alertas de tipo bogus. Para estas pruebas se utiliza la base

de datos ALeRCE-ZTFv7 y alertas de primeras detecciones no etiquetadas, provenientes del flujo de alertas que genera ZTF cada noche.

La base de datos ALeRCE-ZTFv1 y ALeRCE-ZTFv7, se diferencian por la fecha en que fueron generadas, y por ende en la cantidad de datos por clase que poseen, ya que versiones con números más altos indican que fueron generadas en etapas más avanzadas del estudio de ZTF. La base de datos ALeRCE-ZTFv7 fue generada en 08/2020 y su distribución de muestras por clase es; 21,990 AGN, 29,764 VS, 2312 SN, 9899 asteroides y 17,278 bogus (1980 *bogus del paso 1* y 15,298 *bogus del paso 2*), i.e., se cuenta con 63,965 *inliers*. Se ha de recordar que los bogus han sido etiquetados de forma visual por expertos, y los *bogus del paso 2* son particularmente parecidos a las muestras reales tipo SN, debido a que son bogus etiquetados como SN por versiones iniciales del stamp classifier. Las clases AGN, VS, SN y asteroide son consideradas *inliers*.

Debido a que se busca evidenciar el efecto de la utilización exclusiva de imágenes, es que se utiliza una versión del stamp classifier que en su arquitectura no incluye la concatenación de metadatos, ni subsecuente capa de *batch normalization*, manteniéndose el resto del modelo igual al descrito en la Sección 3.1.4.

El pre-procesamiento, partición de datos y entrenamiento es idéntico al presentado en la Sección 3.1 para el stamp classifier. La única diferencia radica en que se utiliza la base de datos ALeRCE-ZTFv7, usando conjuntos de test y validación con 300 muestras por clase, como se utilizan las clases SN, VS, AGN, asteroide y bogus, se tiene un total de 1,500 muestras tanto en el conjunto de validación como en el de test. La utilización de una mayor cantidad de muestras en conjuntos de test y validación respecto de experimentos anteriores, radica en que en la base de datos ALeRCE-ZTFv7 cuenta con más objetos de la clase SN. Se toma la precaución de utilizar un conjunto de test que no presenta muestras en común con ningún conjunto de las bases de datos utilizadas en secciones anteriores (ALeRCE-ZTFv5 y small-ALeRCE-ZTFv1).

### 3.3.1. Procedimiento

Para evidenciar la efectividad de etiquetar bogus con GeoTransform sobre el desempeño del stamp classifier sin metadatos, es que se entrena este modelo utilizando bogus obtenidos a partir de diferentes escenarios, los que simulan diferentes etapas de un estudio astronómico. Se asume que en todos los casos se tiene acceso a alertas etiquetadas de objetos astronómicos reales (obtenidas por *cross-match*), las que corresponden a muestras de las clases AGN, SN, VS y asteroide de la base de datos ALeRCE-ZTFv7, mientras que los objetos de tipo bogus dependen de cada uno de los siguientes escenarios:

1. **Comienzo de un estudio astronómico - Bogus de GeoTransform:** No se cuenta con ninguna alerta tipo bogus etiquetada por expertos, por lo cual se utilizan versiones mejoradas de GeoTransform para etiquetar bogus de forma automática, este escenario y sus bogus reciben el nombre de *bogus de GeoTranform*.
2. **Etapa temprana - Bogus tempranos:** Se tiene acceso a una pequeña porción de bogus etiquetados por expertos. Se utilizan los *bogus del paso 1* (1907 bogus al descartar los del conjunto de test), que corresponden a bogus identificables como tal a simple

vista, y proporcionados por ZTF. Este escenario y sus bogus reciben el nombre de *bogus tempranos*.

3. **Etapa avanzada - Bogus avanzados:** Se utilizan los 16,870 bogus disponibles en la base de datos ALeRCE-ZTFv7 (sin incluir conjunto de test), lo que incluye tanto *bogus del paso 1* como *bogus del paso 2*, estos últimos consideran el etiquetado por expertos de ALeRCE durante más de 8 meses. Este escenario y sus bogus reciben el nombre de *bogus avanzados*.

En el escenario de bogus de GeoTransform, este algoritmo es entrenado utilizando un subconjunto de los *inliers* de entrenamiento y validación de ALeRCE-ZTFv7. A estos subconjuntos de *inliers* se les denomina base de datos small-ALeRCE-ZTFv7 y está compuesto por un conjunto de 7000 inliers de entrenamiento y 1000 inliers de validación, extraídos de forma aleatoria desde los conjuntos de entrenamiento y validación de la base de datos ALeRCE-ZTFv7.

Utilizando small-ALeRCE-ZTFv7, GeoTransform se entrena siguiendo el esquema de entrenamiento con detención temprana, descrito en la Sección 3.2.5.2. Luego, se extrae un conjunto de 50,000 primeras alertas sin etiquetar desde 10 días seguidos del flujo de alertas de ZTF, conjunto que será referenciado como *conjunto de 50,000 primeras alertas* en el resto del documento. Se usa GeoTransform para clasificar como inliers o bogus a todas las muestras de este conjunto. Finalmente, se toman las muestras clasificadas como bogus por GeoTransform y se incluyen en el conjunto de entrenamiento y validación de ALeRCE-ZTFv7, para entrenar el stamp classifier sin metadatos.

Cuando se utiliza GeoTransform para etiquetar muestras bogus, se utiliza GeoTransform72, GeoTransform99 y GeoTransform99 con la selección de transformaciones que obtiene los mejores resultados tanto en small-HiTS como en small-ALeRCE-ZTFv1.

Para la clasificación de bogus con GeoTransform se utilizan los 3 modelos mencionados en el párrafo anterior, debido a la complicación de estipular uno como el mejor de forma definitiva, sin usar un conjunto de test que contenga bogus.

Se espera que los casos donde se entrena el stamp classifier con bogus de GeoTransform presenten mejores métricas en el conjunto de test que aquella que sólo involucra bogus tempranos. Por otro lado, no se espera superar el escenario donde se usan bogus avanzados al entrenar, ya que este es el caso ideal donde una gran cantidad de bogus son identificadas a mano.

### 3.3.2. Evaluación

Cada forma distinta de entrenar el stamp classifier sin metadatos, ya sea con bogus de GeoTransform, bogus tempranos o bogus avanzados, es evaluada utilizando los 1500 datos de test definidos para ALeRCE-ZTFv7. Se reporta el promedio de 10 instancias de entrenamiento de métricas de *accuracy* y matrices de confusión promedio.

# Capítulo 4

# Resultados y análisis

Al igual que en capítulos anteriores, se divide el trabajo presentado en 3 grandes secciones; clasificación temprana de objetos astronómicos, detección de anomalías tipo bogus y la combinación de ambas en detección automática de bogus para la clasificación de objetos astronómicos.

## 4.1. Clasificación temprana de objetos astronómicos

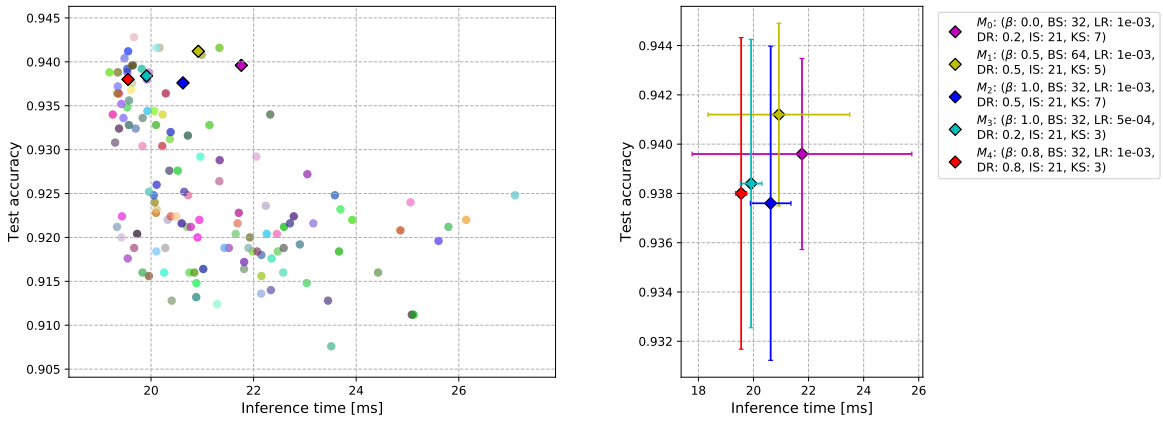
A continuación se describen los resultados de utilizar el stamp classifier para la clasificación de 5 clases de objetos astronómicos, basada en la primera alerta del flujo de datos de ZTF. Para el entrenamiento y validación de modelos se utiliza la base de datos ALeRCE-ZTFv5, definida en la Sección 3.1.1 de la Metodología.

### 4.1.1. Búsqueda aleatoria de hiperparámetros

Para la búsqueda aleatoria de hiperparámetros del stamp classifier se entrenan modelos con 133 diferentes combinaciones de hiperparámetros muestreados desde la Tabla 3.2. Se entrenan 5 modelos por cada combinación de hiperparámetros, utilizando los conjuntos de entrenamiento y validación de la base de datos ALeRCE-ZTFv5. Para cada modelo entrenado se mide su *accuracy* en el conjunto de validación y test, además de medir su tiempo de inferencia para una alerta. Debido que es una búsqueda de hiperparámetros los todas las decisiones de selección de modelos se toman en base al desempeño de modelos en el conjunto de validación, el conjunto de test sólo se usa para visualizar los resultados finales. En la Figura 4.1 se muestran los resultados de la búsqueda de hiperparámetros, se usa la abreviación de términos utilizada en la Tabla 3.2, donde BS es *batch size*, LR es *learning rate*, DR es *dropout rate*, IS es *image size* y KS es *kernel size*. El proceso de entrenamiento toma $\sim$3 días de entrenamiento continuo, al ser realizado en paralelo en 5 GPUs NVIDIA GTX 1080Ti. En esta sección, cada vez que se refiera a *accuracy* o tiempo de inferencia de un modelo se refiere al promedio calculado sobre 5 modelos entrenados con los mismos hiperparámetros.

La selección de la mejor combinación de hiperparámetros se realiza al tomar los 5 modelos con más alto *accuracy* en validación, y de aquellos modelos se escoge el único modelo con

$\beta = 0.5$, ya que según astrónomos con este valor de $\beta$ se obtiene la mejor interpretabilidad de las probabilidades predichas por el modelo. En la Figura 4.1a se grafica el *accuracy* de test versus tiempo de inferencia para las 133 combinaciones de hiperparámetros probadas, en forma de diamante se observan los 5 modelos con más alto *accuracy* en validación. En la Figura 4.1b se observan en detalle estos 5 modelos con el más alto *accuracy* en validación, cada vez que se refiera a estos 5 modelos se habla de *top*-5. En las figuras mencionadas, mientras más cerca de la esquina superior izquierda esté un modelo, mejor es dicho modelo.




(a) (b)

Figura 4.1: Desempeño de los 133 modelos de la búsqueda aleatoria de hiperparámetros. Para cada modelo, los resultados consideran 5 entrenamientos y sus respectivas evaluaciones en el conjunto de test. Todas las decisiones de selección de modelos se realizan utilizando el desempeño sobre el conjunto de validación, en estas figuras se muestran resultados sobre conjunto de test para evidenciar el desempeño real de los modelos. (a) *Accuracy* de test versus tiempo de inferencia, donde cada punto es un modelo con diferentes hiperparámetros, mientras más cerca de la esquina superior izquierda esté un modelo, mejor es su desempeño. Modelos representados por diamantes corresponden a los 5 modelos con mejor *accuracy* de validación. (b) *Accuracy* de test versus tiempo de inferencia para los modelos representados como diamantes en (a), cada modelo incluye sus barras de error correspondientes a una desviación estándar, y cada modelo es denotado como $M_{\mathrm{i}}$, donde i corresponde al *ranking* de su *accuracy* de validación respecto a todos los 133 modelos de (a). Figura obtenida de [2].

Resultados detallados y los hiperparámetros utilizados para los modelos *top*-5 pueden verse en la tabla 4.1, donde se nombra a cada modelo como $M_{\mathrm{i}}$ de acuerdo a su *ranking* de *accuracy* de validación respecto a los 133 modelos probados en la búsqueda aleatoria de hiperparámetros. En esta tabla se muestra que se realiza un test de hipótesis de Welch, entre el modelo con más alto ($M_0$) y más bajo ($M_4$) *accuracy* de validación entre el *top*-5, obteniéndose un p-valor de 0.594, lo que significa que no existe una diferencia estadística significativa entre los *accuracies* de los modelos top-5. Este test de hipótesis también se realiza para el tiempo de inferencia entre $M_0$ y $M_4$, donde se obtiene un p-valor de 0.330, osea que tampoco hay diferencia estadística entre los modelos. Debido a la falta de diferencias estadísticamente significativas entre los modelos *top*-5, es que se escoge el modelo $M_1$ para el resto de los experimentos y para ser implementado en ALeRCE, ya que tiene un $\beta = 0.5$, y de acuerdo a los astrónomos, este valor de $\beta$ entrega el rango de probabilidades predichas con mejor interpretabilidad. Ejemplos de cómo distribuyen las probabilidades predichas para diferentes

valores de $\beta$ pueden observarse en la Figura 4.7. El modelo escogido $M_1$ tiene un *accuracy* en validación de 0.950±0.003, *accuracy* en test de 0.941±0.004 y tiempo de inferencia de 20.5±2.6 [ms]. En la tabla 4.1, se pueden ver subrayadas las medidas del modelo escogido $M_1$, mientras que en negrita aparecen las mediciones con la media más alta de cada columna. Por mera casualidad $M_1$ presenta el *accuracy* en test con media más alta.

Tabla 4.1: Desempeño de los 5 modelos con más alto accuracy en validación en la búsqueda aleatoria de hiperparámetros, ordenados desde $M_0$ a $M_4$. El modelo $M_1$ es escogido para el resto de los experimentos e implementación en ALeRCE, por presentar las probabilidades de predicción más interpretables con $\beta = 0.5$, de acuerdo a astrónomos. Las métricas del modelo $M_1$ están subrayadas, mientras que las métricas en negrita son las con media más alta de sus respectivas columnas.

| Nombre del modelo | Hiperparámetros del modelo | *Accuracy* de validación | *Accuracy* de Test | Tiempo de Inferencia [ms] |
|---|---|---|---|---|
| $M_0$ | $\beta$: 0, BS: 32, LR: 1e-03, DR: 0.2, IS: 21, KS: 7 | **0.950±0.003** | 0.940±0.004 | 21.8±4.0 |
| $M_1$ | $\beta$: 0.5, BS: 64, LR: 1e-03, DR: 0.5, IS: 21, KS: 5 | 0.950±0.005 | **0.941±0.004** | 20.9±2.6 |
| $M_2$ | $\beta$: 1.0, BS: 32, LR: 1e-03, DR: 0.5, IS: 21, KS: 7 | 0.949±0.002 | 0.938±0.006 | 20.6±0.7 |
| $M_3$ | $\beta$: 1.0, BS: 32, LR: 5e-04, DR: 0.2, IS: 21, KS: 3 | 0.948±0.003 | 0.938±0.006 | 19.9±0.4 |
| $M_4$ | $\beta$: 0.8, BS: 32, LR: 1e-03, DR: 0.8, IS: 21, KS: 3 | 0.949±0.003 | 0.938±0.006 | **19.6±0.2** |
| Welch's t-test p-valor $M_0$ v/s $M_4$ \| $M_1$ v/s $M_4$ \| $M_0$ v/s $M_4$ | | 0.594 | 0.364 | 0.330 |

### 4.1.2. Desempeño del modelo

Los resultados presentados a continuación utilizan el modelo $M_1$ escogido en la búsqueda de hiperparámetros de la sección anterior, lo que considera tamaño de batch 64, tasa de aprendizaje 1e-3, tasa de *dropout* de 0.5, tamaño de imágenes de $21 \times 21$, tamaño de kernel 5 para las capas convolucionales y $\beta = 0.5$ para la regularización de entropía. Las métricas y matrices de confusión presentadas consideran el promedio y desviación estándar para 5 entrenamientos de cada modelo. Dado que los conjuntos de datos de validación y test están balanceados, se muestra como principal métrica el *accuracy* del modelo. Como se observa en la Tabla 4.1, $M_1$ tiene un *accuracy* en validación de 0.950±0.003, *accuracy* en test de 0.941±0.004 y tiempo de inferencia de 20.5±2.6 [ms].

El modelo escogido $M_1$ será referido como stamp classifier en el resto del documento. La Figura 4.2 muestra su matriz de confusión promedio en el conjunto de test.

En la matriz de confusión de la Figura 4.2 se aprecia como el modelo presenta un recall de $0.87 \pm 0.01$ para la clase SN, con una tasa de falsos positivos de apenas $5 \pm 2\,\%$. El desempeño del stamp classifier en la clase SN es de especial interés, ya que su principal uso en ALeRCE es la detección y descubrimiento de nuevas SNe.

La evolución del *accuracy* y función de costo a lo largo del proceso de entrenamiento, para los conjuntos de validación y entrenamiento de una única instancia del stamp classifier, pueden apreciarse en la Figura 4.3. En la Figura 4.3a se tiene que el menor valor de función de costo se alcanza en la iteración 6310, tras lo cual el modelo comienza a sobre-ajustarse al conjunto de entrenamiento. Puede resultar curioso que la función de costo tenga un valor menor a cero, esto se debe a la utilización de la regularización de entropía en la ecuación de la función de costo (ecuación 3.1). Del entrenamiento de 5 instancias del stamp classifier,

se puede decir que 30,000 iteraciones demoran 9 minutos y 20 segundos en promedio (al realizar el entrenamiento en una GPU RTX2080Ti), obteniéndose el mejor modelo entorno a la iteración 6000.

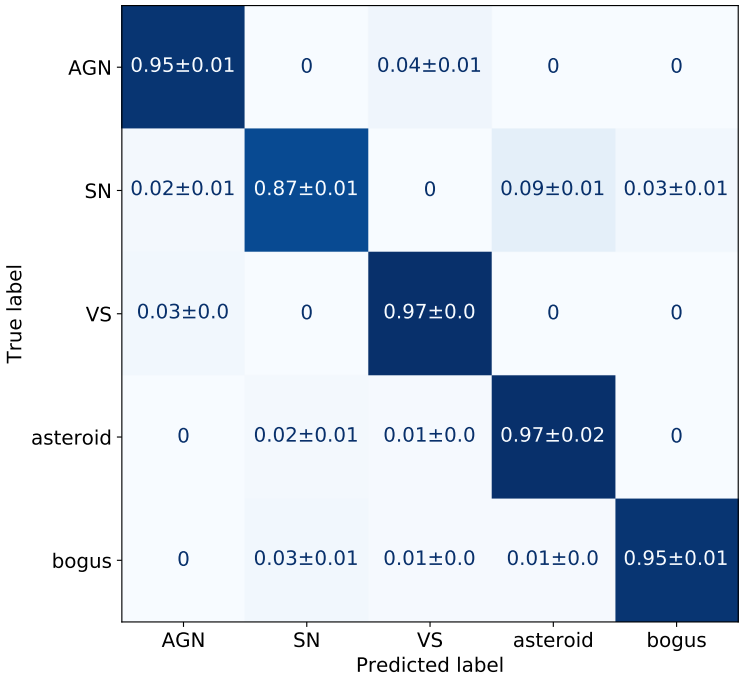
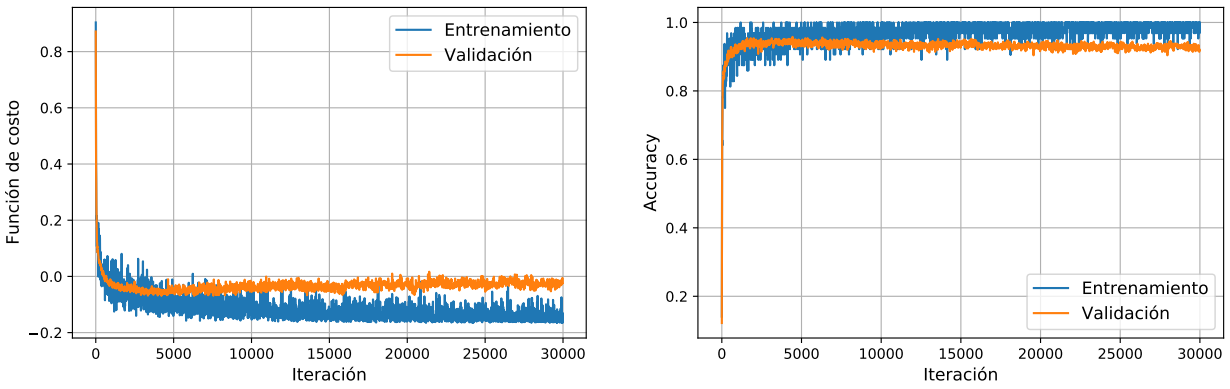



Figura 4.2: Matriz de confusión del stamp classifier. Se muestra el resultado promedio de 5 modelos evaluados sobre el conjunto de test. Figura obtenida de [2].




(a) (b)

Figura 4.3: Curvas de función de costo (a) y *accuracy* (b) para el conjunto de entrenamiento y validación a lo largo del entrenamiento de una instancia del stamp classifier. (a) Curvas de función de costo. (b) Curvas de *accuracy*.

### 4.1.3. Evaluación de predicciones

Al visualizar las predicciones realizadas por una instancia del stamp classifier, se observa que los objetos clasificados como SN presentan características visuales en sus estampas como las que se describen en la Sección 2.1.4 del marco teórico; se observan galaxias anfitrionas en

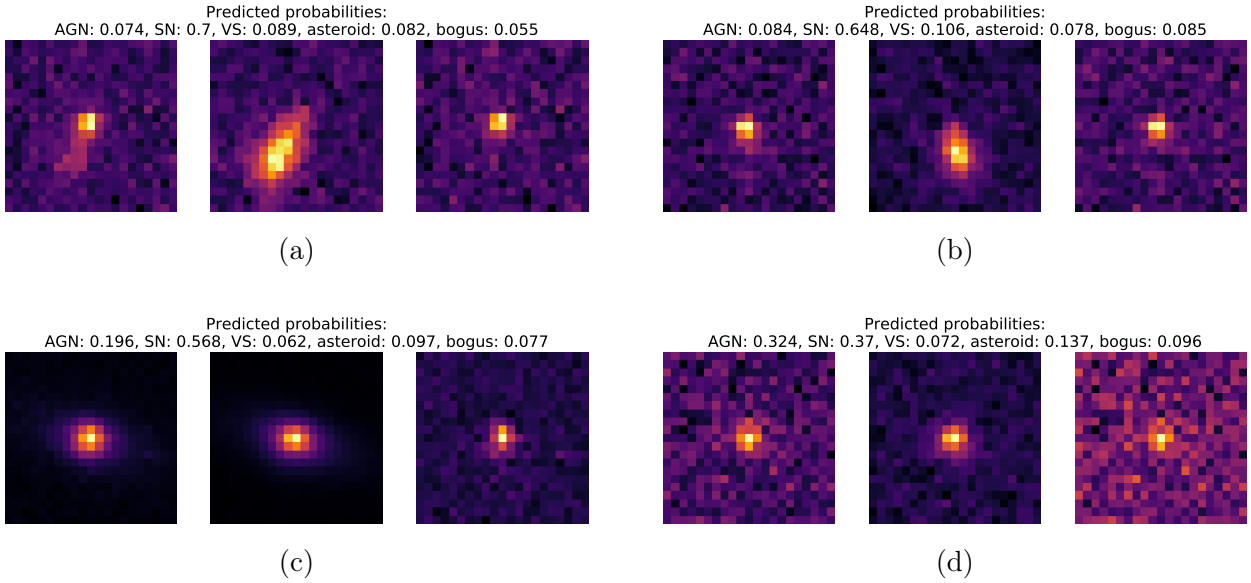



Figura 4.4: SN correctamente clasificadas, donde se muestra el vector de probabilidades (5 valores) predicho por el stamp classifier. Cada tripleta de imágenes muestra las estampas en el orden ciencia, referencia y diferencia. (a) y (b) muestran ejemplos donde se aprecia claramente la galaxia anfitriona en las imágenes, siendo buenos candidatos a ser reportados como SNe reales. (c) y (d) son ejemplos donde la segunda probabilidad más grande es AGN, ya que la fuente luminosa ocurre justo al centro de la galaxia anfitriona. Figura obtenida de [2].

las alertas. En la Figura 4.4 se muestran las tripletas de imágenes correspondientes a objetos correctamente clasificados como SNe por el stamp classifier. En los ejemplos (a) y (b) se aprecia una galaxia anfitriona en la imagen de referencia (imagen al centro de las tripletas de imágenes), mientras que para el ejemplo (a) también aparece la galaxia anfitriona en la imagen de ciencia (imagen más a la izquierda de la tripleta de imñagenes). La presencia de galaxias anfitrionas corresponde a un buen indicador de que se está en presencia de una SN. Para los ejemplos (c) y (d) de la Figura 4.4 la probabilidad de SN tiene poca confidencia, i.e. hay otras clases con gran porcentaje de probabilidad, en ambos casos la segunda clase con mayor probabilidad es AGN, lo que se atribuye a que la fuente luminosa está cerca del centro de la galaxia anfitriona, característica de los objetos de la clase AGN.

Por otro lado, en la Figura 4.5, se pueden observar 4 ejemplos identificados como SNe reales por TNS, las cuales son incorrectamente clasificadas como otros objetos por el stamp classifier. Los ejemplos (a), (b) y (c) de la Figura 4.5 son SNe clasificadas como asteroides, mientras que el ejemplo (d) es clasificado como bogus. Todos los casos presentan confidencias bajas en la clase asignada con mayor probabilidad, y en cada caso la segunda clase con mayor probabilidad es SN. Entiéndase por confidencia baja una probabilidad predicha cercana a 0, y por alta confidencia una probabilidad predicha cercana a 1.

### 4.1.4. Efectos de utilizar metadatos

Para medir los efectos de utilizar metadatos sobre el desempeño del stamp classifier es que se calcula la matriz de confusión de la Figura 4.6. Al no utilizar metadatos como carac-

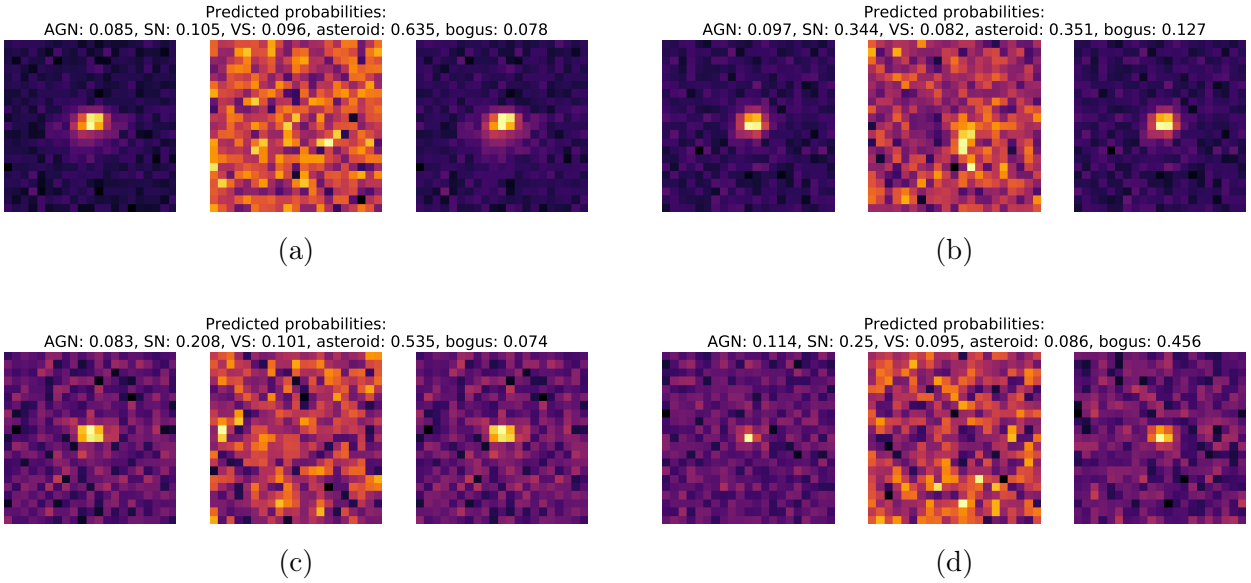



Figura 4.5: Ejemplos de SNe incorrectamente clasificadas como otras clases por el stamp classifier, junto las probabilidades predichas para cada caso. (a), (b) y (c), corresponden a asteroides clasificados como SNe. (d) es una SNe clasificada como una alerta bogus. En todas las SNe erróneamente clasificados como otras clases se nota la ausencia de una galaxia anfitriona. Figura obtenida de [2].

terísticas de entrada en las capas *fully connected* del stamp classifier se logra un *accuracy* promedio de $0.883 \pm 0.006$, con un recall de $0.8 \pm 0.02$ para la clase SN, la que cuenta con una tasa de falsos positivos de $10 \pm 4\,\%$.

Una comparación directa entre el desempeño del stamp classifier con y sin metadatos puede apreciarse en la Tabla 4.2, donde se observa como la utilización de metadatos aumenta el desempeño de todas las métricas para el stamp classifier, además de reducir su desviación estándar. Lo mismo sucede al comparar las matrices de confusión con metadatos de la Figura 4.2 y sin metadatos de la Figura 4.6, donde el desempeño para todas las clases mejora al utilizar metadatos.

Tabla 4.2: Comparación de desempeño entre stamp classifier con metadatos y sin metadatos como características de entrada. En negrita se marca el mejor resultado de cada columna.

| | *Accuracy* | Recall de SN | Falsos positivos de SN |
|---|---|---|---|
| Con metadatos | $\mathbf{94.1 \pm 0.4\,\%}$ | $\mathbf{87 \pm 1\,\%}$ | $\mathbf{5 \pm 2\,\%}$ |
| Sin metadatos | $88.3 \pm 0.6\,\%$ | $80 \pm 2\,\%$ | $10 \pm 4\,\%$ |

Resultados adicionales sobre cuales son los metadatos más importantes para el desempeño del stamp classifier, son estudiados en el artículo publicado [2].

### 4.1.5. Regularización de entropía

Los efectos del término regularizador de entropía en la función de costo al entrenar el stamp classifier pueden verse en la Figura 4.7. Se aprecia con claridad el efecto que tiene

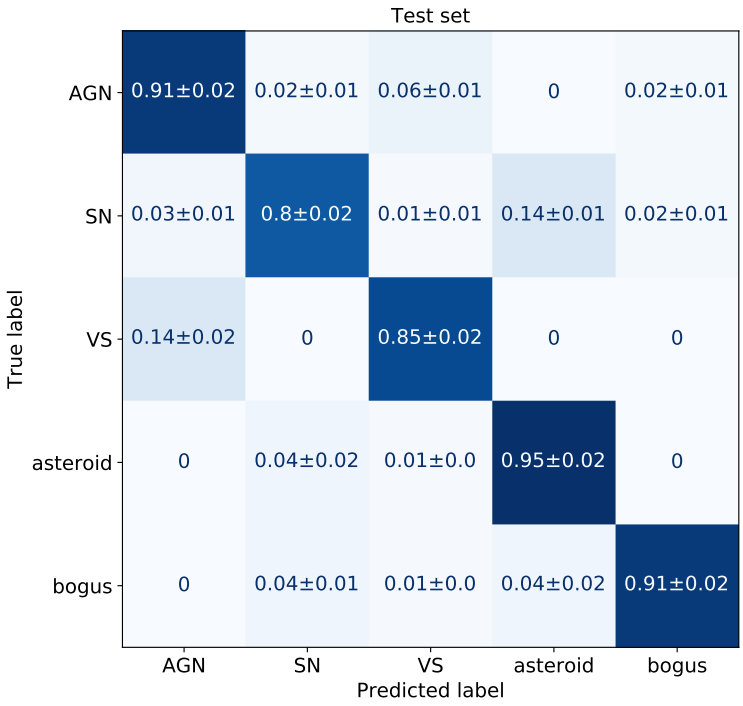



Figura 4.6: Matriz de confusión promedio sobre el conjunto de test, al entrenar 5 instancias del stamp classifier sin los metadatos de las alertas como características de entrada, i.e., utilizando sólo las imágenes. Figura obtenida de [2].

variar la influencia de la regularización de entropía al modificar $\beta$, donde las distribuciones de probabilidades predichas para cada clase cambian fuertemente. Este efecto se debe a que la regularización de entropía en la función de costo de la ecuación 3.1 actúa sobre valores que corresponden a la esperanza del logaritmo de las probabilidades (*log-odds* en inglés).

En la primera fila de la Figura 4.7 se predice con un modelo entrenado con $\beta = 0$, i.e., sin regularización de entropía, lo que hace que las probabilidades predichas se encuentren saturadas en valores 0 o 1. Esta saturación de la probabilidades predichas dificulta a los astrónomos el análisis de alertas donde no se tenga claridad de la clase a la que pertenecen. El mapeo a niveles de alta confidencia se ve suavizado al aumentar el valor de $\beta$, al utilizar $\beta = 0.5$ se observa cómo las probabilidades predichas se esparcen a lo largo del rango de probabilidades, dando espacio para que el modelo indique a que otras clases puede parecerse una alerta. Mientras más grande es $\beta$ más se desplazan las distribuciones predichas a valores de baja confidencia.

Al modificar $\beta$ se generan variaciones despreciables en el desempeño de los modelos, ya que el orden de las clases predichas se mantiene en la mayoría de los casos, variando solo la probabilidad asignada a cada una, donde la confidencia de la clase predicha baja si $\beta$ aumenta.

La utilización de regularización de entropía genera diferencias notables en cómo se distribuyen las probabilidades del modelo. De acuerdo a astrónomos, utilizar $\beta = 0.5$ ha probado ser útil para analizar las clasificaciones, al obtenerse mejor interpretabilidad de que tan confiables son las clasificaciones realizadas.

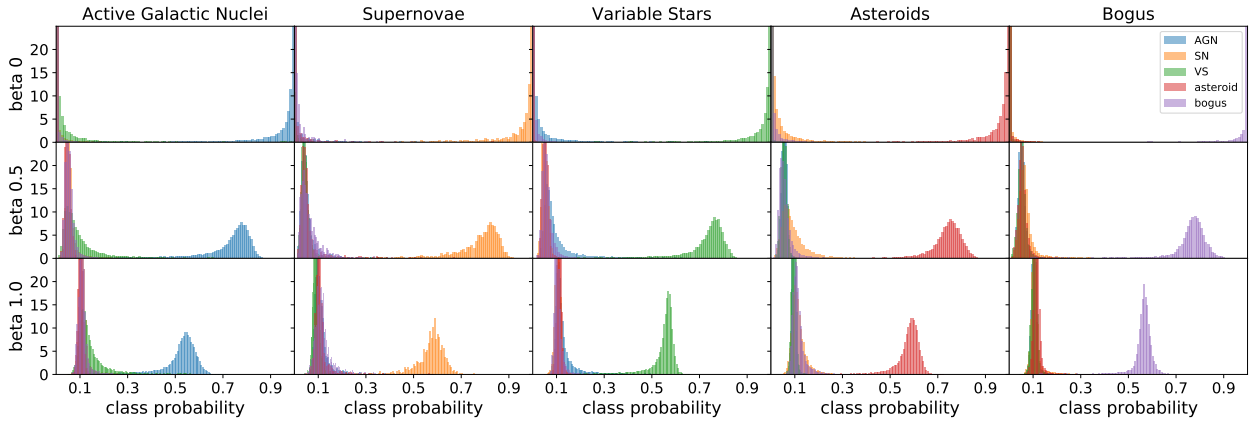



Figura 4.7: Se aprecian las distribuciones de probabilidades para las diferentes clases del stamp clasifier al ser entrenado con valores de $\beta = \{0, 0.5, 1.0\}$, parámetro que controla la regularización de entropía en la función de costo durante el entrenamiento. Al utilizar $\beta = 0$ se tiene un modelo sin regularización de entropía donde la mayoría de las probabilidades predichas se encuentran saturadas en valores 0 o 1. Al aumentar el valor de $\beta$ a 0.5 o 1.0, se obtienen penalizaciones más grandes a la baja entropía de las probabilidades predichas, por lo cual estas tienden a esparcirse a lo largo de las diferentes clases. Figura obtenida de [2].

### 4.1.6. Supernovae Hunter

A continuación se presentan resultados que no son de autoría exclusiva del tesista, pero son resultados importantes que surgen en consecuencia del trabajo realizado y la creación del stamp classifier.

El objetivo principal del stamp classifier es la detección de SNe, de tal forma de ser un filtro para que astrónomos puedan visualizar los candidatos más prometedores y reportarlos para *follow-up* con telescopios especializados capaces de confirmar si realmente el objeto detectado es una SNe. El stamp classifier entrega la posibilidad de descubrir SNe nunca antes vistas.

Para una cómoda visualización de las alertas detectadas por el stamp classifier, es que el equipo de ALeRCE desarrolló el *SN Hunter* (`https://snhunter.alerce.online/`). El SN Hunter es una herramienta en forma de una aplicación web donde astronómos pueden visualizar las clasificaciones del stamp classifier sobre el flujo de alertas de ZTF de cada noche, donde se muestran las 100 SNe con mayor probabilidad predicha de la última noche. Utilizando esta herramienta, un grupo de astrónomos de ALeRCE revisa día a día los mejores candidatos a SNe según el stamp classifier y decide cuales son buenos candidatos como para ser reportados para *follow-up* a TNS y verificar si es que efectivamente son nuevas SNe descubiertas por ALeRCE. Detalles del SN Hunter, y del proceso realizado por astrónomos para seleccionar candidatos a reportar a TNS, se pueden encontrar en [2].

Entre el 26 de Junio del 2019 y el 28 de Febrero del 2021, utilizando el stamp classifier y otras herramientas, ALeRCE ha reportado más de 6846 candidatos a SN a TNS (11.8 candidatos en promedio por día), siendo 971 confirmados espectroscópicamente como nuevas SNe por TNS. En la Tabla 4.3 pueden observarse 995 objetos reportados por ALeRCE y observados por TNS, con sus clases específicas asignadas. La mayoría de la SN observadas por TNS corresponden al tipo SN Ia y SN II. En la Tabla 4.3 se aprecian 24 objetos que

fueron erróneamente reportados como SNe; 5 *tidal disruption event* (TDE), 5 galaxias, 4 Novas, 2 otras, 2 *cataclysmic variables* (CV), 2 AGNs, 2 no identificados, 1 VS y 1 *M dwarf*, por lo cual se puede decir que de los candidatos observados por TNS, ALeRCE presenta un 2.4 % de contaminación de objetos no-SN en sus reportes confirmados por TNS.

Tabla 4.3: Objetos que han sido descubiertos por ALeRCE y espectroscópicamente confirmados por TNS. De los 995 objetos observados por TNS 971 corresponde a SNe, mientras 24 fueron confirmados como no-SN, i.e. son reportes de falsas SNe, estos objetos comprenden 5 *tidal disruption event* (TDE), 5 galaxias, 4 Novas, 2 otras, 2 *cataclysmic variables* (CV), 2 AGNs, 2 no identificados, 1 VS y 1 *M dwarf*.

| Clase confirmada | Número de candidatos observados espectroscópicamente |
|---|---|
| SN Ia | 676 |
| SN II | 148 |
| SN Ic | 24 |
| SN Ia-91T-like | 22 |
| SN IIn | 21 |
| SN IIP | 16 |
| SN Ib | 14 |
| SN IIb | 13 |
| SN Ic-BL | 10 |
| TDE | 5 |
| Galaxia | 5 |
| Nova | 4 |
| SN Ia-pec | 4 |
| SN Iax[02cx-like] | 4 |
| SN I | 3 |
| SN Ia-91bg-like | 3 |
| SN | 3 |
| SN Ib/c | 3 |
| SLSN-II | 3 |
| Otras | 2 |
| CV | 2 |
| No identificados | 2 |
| AGN | 2 |
| SN Ib-pec | 1 |
| VS | 1 |
| SN Ibn | 1 |
| M dwarf | 1 |
| SLSN-I | 1 |
| SN Icn | 1 |

Otra cualidad destacable del stamp classifier es su detección con una sola alerta, por lo cual permite identificación rápida de SN, las cuales son reportadas por los astrónomos en las primeras 10 a 24 horas desde su detección por ZTF. Estudios detallados y estadísticas de los tiempos de reporte de alertas utilizando el stamp classifier y el SN Hunter se pueden encontrar en [2].

Se ha de mencionar que el SN Hunter también ha permitido identificar alertas bogus que son erróneamente clasificadas como SN por el stamp classifier, lo que ha hecho posible construir una gran base de datos de alertas bogus etiquetadas por expertos.

## 4.2. Detección de anomalías tipo bogus

A continuación se presentan los resultados relacionados al segundo objetivo general de esta tesis, el que comprende la mejora del algoritmo GeoTransform para detección automática de artefactos astronómicos tipo bogus, basada en las imágenes de alertas astronómicas. Una versión preliminar de estos resultados fue publicada en [4], el cual es extendido y profundizado en esta tesis, particularmente la sección de selección de transformaciones para GeoTransform.

Para evaluar los modelos de detección de anomalías se usan las bases de datos small-HiTS y small-ALeRCE-ZTFv1. La utilización de múltiples bases de datos se sustenta en que los métodos de detección de anomalías no pueden ser validados utilizando anomalías, estas sólo están disponibles al momento de evaluar el modelo en el conjunto de test, por lo cual se desea evitar un sobre-ajuste de estos a los conjuntos de test de las diferentes bases de datos. Las bases de datos utilizadas son versiones reducidas, ya que GeoTransform no escala bien en recursos computacionales consumidos al aumentar la cantidad de datos.

Durante el entrenamiento de los modelos, sólo se tiene acceso a la clase *inlier* (objetos astronómicos reales). Al comparar modelos, se utiliza principalmente la métrica de AUROC sobre un conjunto de test balanceado, el que contiene *inliers* y bogus. Los resultados mostrados en cada tabla corresponden al promedio y desviación estándar de 10 instancias de modelos entrenados con pesos iniciales aleatorios.

### 4.2.1. Esquemas de entrenamiento

Se compara el desempeño de GeoTransform bajo diferentes esquemas de entrenamiento; el original propuesto en [21] y utilización de detención temprana. Estos esquemas serán referidos como *original* y *detención temprana*, respectivamente. Como se mencionó en la Sección 3.2.5 de la Metodología, se ha propuesto de forma preliminar un tercer esquema de entrenamiento que permite entrenar GeoTransform sobre bases de datos arbitrariamente grandes, detalles y resultados preliminares para este esquema de entrenamiento pueden ser consultados en la Sección B.1 del Apéndice.

En el esquema original de entrenamiento de GeoTransform [21] se usa un número fijo de épocas, con el objetivo de utilizar un esquema parecido a otros métodos de la literatura que usan 200 épocas de entrenamiento. Debido a que GeoTransform hace uso de un conjunto de datos auto-etiquetado $S_{\mathcal{T}}$ que es $\mathcal{T}$ veces más grande que los originales, se ajusta la cantidad de iteraciones a $200/\mathcal{T}$ épocas. Ya que existe una correlación directa entre que tan bien son clasificadas las transformaciones y la calidad del puntaje de normalidad, es que se proponen estrategias que eviten el sobre-ajuste del modelo al conjunto de entrenamiento. Se utiliza un conjunto de validación compuesto únicamente por *inliers*, con el que se implementa detención temprana del entrenamiento. En el resto del documento, cada vez que se entrena un modelo referido como GeoTransform sobre las bases reducidas (small-HiTS y small-ALeRCE-ZTFv1), se utiliza detención temprana con *patience* 0, evaluando el modelo en el conjunto de validación

al final de cada época. La única excepción a esta regla es el modelo que se denominará *original GeoTransform72*, el cual se entrena con el esquema original de entrenamiento de $200/\mathcal{T}$ épocas. La Tabla 4.4 muestra el AUROC obtenido sobre las bases de datos small-HiTS y small-ALeRCE-ZTFv1 al utilizar el esquema de entrenamiento original y la utilización de detención temprana.

Tabla 4.4: Comparación de esquemas de entrenamiento para GeoTransform. Se compara el esquema original de entrenar durante $200/\mathcal{T}$ épocas fijas, contra utilizar detención temprana.

| Modelo | Esquema de entrenamiento | small-HiTS AUROC | small-ALeRCE-ZTFv1 AUROC |
|---|---|---|---|
| Original GeoTransform72 [21] | $200/\mathcal{T}$ épocas | 98.60±0.23 | 85.63±1.48 |
| GeoTransform72 | Detención temprana | **98.78±0.26** | **87.33±1.24** |

En la Tabla 4.4 se aprecia como para ambas bases de datos, el desempeño de GeoTranform72 mejora al utilizar detención temprana.

### 4.2.2. Nuevas transformaciones

Se incluyen transformaciones basadas en kernels Gaussiano y Laplaciano a GeoTransform, con el objetivo de aprovechar conocimiento a-priori de la morfología de muestras tipo bogus. Debido a que una operación de kernel puede ser aplicada o no, esto agrega dos posibles estados a la composición de transformaciones, e.g. como GeoTransform originalmente tiene 72 transformaciones (GeoTransform72), incluir una nueva transformación con dos estados, significa que se duplica el número de transformaciones a 144, para que se incluyan todas las posibles combinaciones del conjunto actual de transformaciones. Para evitar el costo computacional de generar demasiadas transformaciones, se usa principalmente una configuración simplificada al aplicar las transformaciones basadas en kernels **sólo** sobre las operaciones de traslación, las que poseen 9 estados posibles. En este caso, incluir una transformación basada en kernel agrega sólo 9 transformaciones, de este modo, para el ejemplo de GeoTransform72, si es que se agrega una transformación basada en kernel a la composición que genera las 72 transformaciones originales, estas aumentan a 81.

Siguiendo el procedimiento anterior, se compara GeoTransform72, con la configuración simplificada de incluir solo kernel Gaussiano (GeoTransform81-G) o sólo kernel Laplaciano (GeoTransform81-L) e incluir ambas operaciones de kernel (GeoTransform99). Para GeoTransform99, 9 transformaciones corresponden a kernel Gaussiano, 9 son de kernel Laplaciano y 9 son la combinación de ambos tipos de kernel. Al agregar kernels Laplaciano o Gaussiano al conjunto de 72 transformaciones se llega a un conjunto de 144 transformaciones, i.e. GeoTransform144-G para kernel Gaussiano y GeoTransform144-L para kernel Laplaciano, mientras que al aplicar ambos kernels se tienen 288 transformaciones (GeoTransform288). Siempre que ambos kernel estén presentes en una transformación, primero se aplica el kernel Gaussiano y luego el Laplaciano.

La Tabla 4.5 muestra el AUROC al utilizar las transformaciones propuestas sobre small-HiTS y small-ALeRCE-ZTFv1. En esta tabla se observa cómo la adición de cualquier trans-

formación de kernel mejora el desempeño base de GeoTranform72 en ambas bases de datos. Los mejores resultados se logran al utilizar kernel Laplaciano y Gaussiano sobre las traslaciones en la configuración con 99 transformaciones (GeoTransform99). Para ambas bases de datos, GeoTransform99 mejora el AUROC de GeoTransform72, donde un t-test de hipótesis de Welch muestra un p-valor menor a $2.9\times10^{-3}$ para small-HiTS y menor a $4.0\times10^{-6}$ para small-ALeRCE-ZTFv1, siendo ambos p-valores indicadores de que las diferencias entre GeoTransform72 y GeoTransform99 son estadísticamente significativas.

Tabla 4.5: Efecto de incluir diferentes transformaciones basadas en kernels en el conjunto original de 72 transformaciones para entrenar GeoTransform.

| Modelo | Transformaciones usadas (# transformaciones) | small-HiTS AUROC | small-ALeRCE-ZTFv1 AUROC |
|---|---|---|---|
| GeoTransform72 | Original (72) | 98.78±0.26 | 87.33±1.24 |
| GeoTransform81-G | Original + Gauss sobre traslaciones (81) | 99.01±0.07 | 89.28±1.30 |
| GeoTransform81-L | Original + Laplace sobre traslaciones (81) | 98.87±0.11 | 88.94±0.96 |
| GeoTransform99 | Original + (Gauss & Laplace) sobre traslaciones (99) | **99.12±0.04** | **90.80±0.61** |
| GeoTransform144-G | Original + Gauss (144) | 98.84±0.15 | 87.58±0.84 |
| GeoTransform144-L | Original + Laplace (144) | 98.85±0.15 | 87.62±1.00 |
| GeoTransform288 | Original + (Gauss & Laplace) (288) | 99.02±0.05 | 89.79±0.67 |
| p-valor de t-test de Welch (72) v/s (99) | | $2.9\times10^{-3}$ | $4.0\times10^{-6}$ |

Debido a los altos costos computacionales de entrenar GeoTransform con más de 99 transformaciones es que en los experimentos siguientes, solo se hacen pruebas sobre GeoTransform99.

### 4.2.3. Selección de transformaciones

A continuación se muestran resultados relacionados a la selección empírica de transformaciones para GeoTransform. Se ha de recordar que las estrategias propuestas en esta sección son de carácter exploratorio, con el objetivo de evidenciar los efectos de seleccionar transformaciones. No se busca dar un veredicto de la mejor estrategia de selección, debido a la falta de un criterio que no utilice el conjunto de test para la toma de esta decisión, y de este modo evitar un sobre-ajuste al conjunto de test. A fin de cuentas se hace necesario un estudio más profundo que permita tomar decisiones sobre los métodos propuestos en esta sección y relacionarlos con algún marco teórico en donde se incluyan sinergías entre transformaciones, como se mencionó en la Sección 3.2.8.3 de la Metodología.

Se ha de recordar que todos los métodos propuestos para selección de transformaciones sólo utilizan *inliers* de los conjuntos de entrenamiento o validación de cada base de datos. El desempeño de entrenar GeoTransform con las transformaciones seleccionadas por cada

método se reporta en términos de AUROC sobre el conjunto de test de small-HiTS y small-ALeRCE-ZTFv1.

La selección empírica comprende la búsqueda de métodos que permitan descartar transformaciones inútiles $T_{inutil}$, definidas bajo tres criterios en la Sección 3.2.8.4:

- $T_{inutil.1}$: Borran toda la información de los datos
- $T_{inutil.2}$: Transformaciones redundantes ya presentes en los datos o en en el conjunto de transformaciones a operar $\mathcal{T}_i$.
- $T_{inutil.3}$: Aquellas que al ser incluidas bajan el desempeño del algoritmo en la tarea final.

Para cada tipo de $T_{inutil}$ se presentan métodos que las buscan y descartan. Para validar la efectividad de estos métodos y su capacidad de descartar las $T_{inutil}$ que buscan, se utiliza small-HiTS y un conjunto de transformaciones $\mathcal{T}_{juguete}$, que incluye transformaciones donde se tiene una fuerte intuición de que son del tipo $T_{inutil}$ para HiTS, además de las transformaciones más representativas de cada tipo, como traslaciones y otras basadas en kernel. Este conjunto de transformaciones recibe el nombre de *conjunto de juguete* $\mathcal{T}_{juguete}$ y contiene 10 transformaciones:

$T_0$: Identidad.

$T_1$: *Flip* vertical; $T_{inutil.2.1}$ para HiTS (HiTS debería ser invariante a *flips*).

$T_2$: Traslación 8 pixeles arriba.

$T_3$: Rotación en 90 grados; $T_{inutil.2.1}$ para HiTS (HiTS debería ser invariante a rotaciones).

$T_4$: Kernel Gaussiano.

$T_5$: Kernel Laplaciano.

$T_6$: Re-ordenamiento aleatorio de pixeles; $T_{inutil.3}$, al no tener una estructura identificable no debería agregar información extra que ayude a la discriminación de anomalías tipo bogus.

$T_7$: Ruido Gaussiano; $T_{inutil.1}$, ya que el valor de cada pixel es reemplazado por un valor muestreado de una distribución $\mathcal{N}(0, 1)$ (destruye la información original).

$T_8$: Multiplicación por 0; $T_{inutil.1}$ (destruye la información original).

$T_9$: *Flip* vertical y Traslación 8 pixeles arriba; $T_{inutil.2.1}$, en HiTS debería ser idéntica a sólo hacer traslación arriba (HiTS debería ser invariante a *flips*).

Pese a que no se tiene certeza absoluta de que HiTS sea invariante a transformaciones como $T_1$, $T_3$ o que se encuentre redundante a $T_9$, debido a que los *inliers* de HiTS son imágenes con fuentes puntuales en su centro, se tienen fuertes intuiciones de que así es, por lo cual si se logran descartar estas transformaciones, se declara el método como útil para encontrar las transformaciones de tipo $T_{inutil}$ que busca. En la Figura 4.8 se pueden ver los efectos de aplicar las 10 transformaciones de juguete sobre un *inlier* de small-HiTS.

Tras utilizar small-HiTS y $\mathcal{T}_{juguete}$ para validar los métodos para seleccionar transformaciones y descartar transformaciones de tipo $T_{inutil}$, se muestran resultados de sus combinaciones en los denominados *flujos de selección*, mostrando resultados sobre el conjunto de juguete, GeoTransform72 y GeoTransform99, tanto para la base small-HiTS como para small-ALeRCE-ZTFv1.

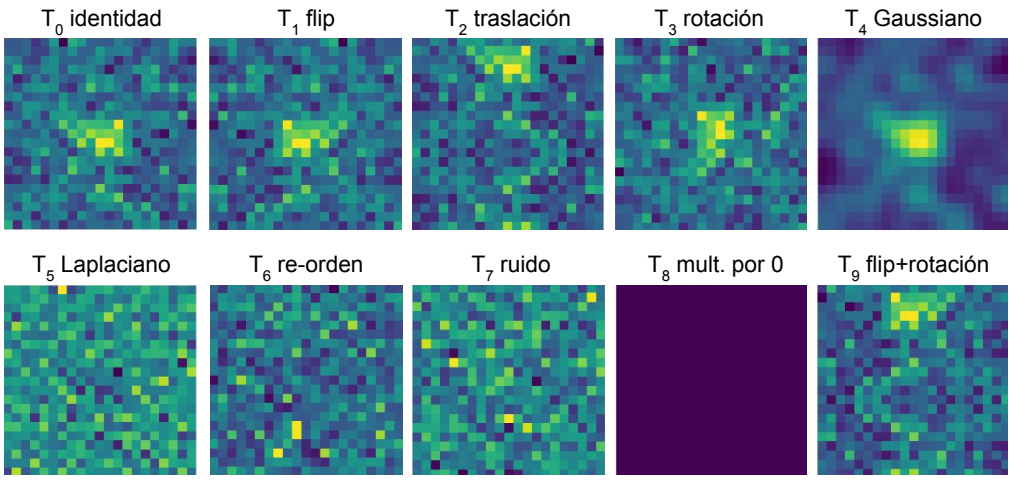



Figura 4.8: Efecto de transformaciones de juguete sobre un *inlier* de HiTS.

En algunos experimentos se reporta el *accuracy* de los modelos, el que se calcula al fijar un umbral $\lambda$ sobre el puntaje de normalidad, de tal forma que 97,7 % de los *inliers* de validación queden a la derecha del umbral. Toda muestra mayor a $\lambda$ es considerada *inlier*, de lo contrario, es clasificada como bogus.

#### 4.2.3.1. Método de selección 1 - Información mutua

Método que busca $T_{inutil.1}$, i.e., transformaciones triviales que borran toda la información de los datos. Este método ($M_1$) mide la razón entre la información mutua del conjunto de datos original $\mathcal{X}$ y su versión transformada $T(\mathcal{X})$, y la auto-MI de la base de datos original, razón expresada por $M_1 = MI(\mathcal{X}; T(\mathcal{X}))/MI(\mathcal{X}; \mathcal{X})$. Si es que esta razón es menor a un umbral $\varepsilon$, se dice que la transformación evaluada es $T_{inutil.1}$. Se usa $\varepsilon = 10^{-2}$ el cual se calcula a partir de medir $M_1$ entre la base de datos CIFAR-10 e imágenes constantes en 0.

La Figura 4.9 muestra el valor de $M_1$ sobre el conjunto de entrenamiento de small-HiTS, para cada transformación de $\mathcal{T}_{juguete}$. Se identifica a la transformación $T_8$ (multiplicar por 0) cómo trivial $T_{inutil.1}$, con $M_1 = 10^{-6}$. Se esperaba que $T_7$ tambien se identificara como $T_{inutil.1}$, ya que reemplaza toda la información original por ruido gausianno, pero esta obtiene $M_1 = 0.39$, con lo cual no se puede descartar como $T_{inutil.1}$. Pese a esto, que $T_7$ obtenga $M_1 = 0.39$ y sea la transformación con menor $M_1$ después de $T_8$ da un indicio de que es una transformación que contiene poca información respecto a las imágenes originales.

#### 4.2.3.2. Método de selección 2.1.a - Matriz de discriminación

Método que busca $T_{inutil.2.1}$, i.e. transformaciones a las que la base de datos es invariante o aquellas que al ser aplicadas generan resultados muy parecidos a otra transformación ya presentes en el conjunto inicial de transformaciones $\mathcal{T}_i$. Este método ($M_{2.1.a}$) mide el grado de similitud entre un par de transformaciones $(T_i, T_j)$ en base a la capacidad de un clasificador de distinguir entre ambas. Si es que al evaluar el clasificador sobre nuevos datos con las mismas transformaciones, este obtiene un *accuracy* entre 49 % y 51 %, se descarta la transformación con más operaciones y el resto se utilizan para entrenar GeoTransform. Se repite este proceso para todos los pares de transformaciones. Se puede visualizar el accurcy obtenido para cada

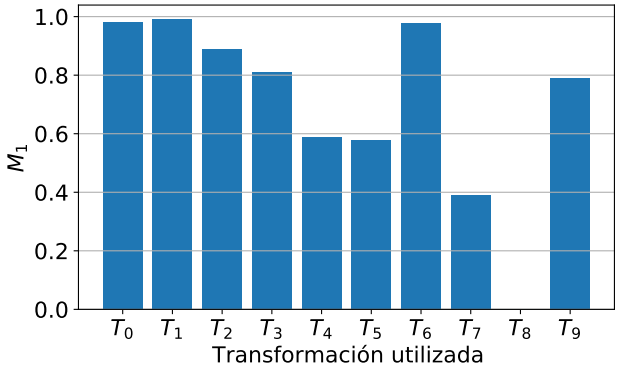



Figura 4.9: Valor de $M_1$ sobre transformaciones de juguete en el conjunto de entrenamiento de small-HiTS.

par de transformaciones en una matriz de discriminación como la de la Figura 4.10.

Se aplica $M_{2.1.a}$ sobre small-HiTS, base de datos cuyos *inliers* se presume poseen invarianza a rotaciones y a *flips*. Se hace una prueba preliminar utilizando un conjunto de transformaciones compuesto de un *flip* vertical y 8 transformaciones de traslación, generándose un total de 18 transformaciones (*flip*, en composición con las 8 traslaciones, en adición a la identidad $T_0$). Para este caso el clasificador no debería poder distinguir entre las versiones que incluyen *flip*, ya que primero se realiza un *flip* de las imágenes, las que deberían ser muy similares a las originales. La matriz de discriminación para las 18 transformaciones se muestra a la izquierda de la Figura 4.10, donde un cuadrado verde indica que la versión con *flip* de una traslación no puede ser discriminada de su contra parte sin el *flip*. Tras eliminar las transformaciones con *flip* en composición con traslaciones, sólo quedan 9 transformaciones *útiles*, las que corresponden a $T_0$ y las 8 traslaciones, confirmándose de este modo la hipótesis de que small-HiTS resulta invariante a *flips*. Al lado derecho de la Figura 4.10 se aprecia la matriz de discriminación para las 9 transformaciones seleccionadas, donde no quedan transformaciones *inútiles* o *redundantes* $T_{inutil.2.1}$. Se obtiene una mejora en AUROC, desde 98.93±0.12 para 18 transformaciones a 99.16±0.13 para 9 transformaciones.

Se prosigue con aplicar $M_{2.1.a}$ sobre las 10 transformaciones del conjunto $\mathcal{T}_{juguete}$ sobre small-HiTS. Se espera encontrar como redundantes a las transformaciones de *flip*, rotación y *flip* en composición con traslación ya presente. Esto se comprueba al observar la matriz de discriminación a la izquierda de la Figura 4.11, donde se indica que las transformaciones de *flip* $T_1$, rotación $T_3$ y *flip* en composición con rotación $T_9$ son redundantes. Particularmente, $T_1$ y $T_3$ se identifican como redundantes con $T_0$ lo que indica que según $M_{2.1.a}$ small-HiTS es invariante a rotaciones y *flips*, mientras que $T_9$ resulta redundante respecto a $T_3$, es decir, una traslación es redundante con un *flip* en composición con una traslación. Pese a que $M_{2.1.a}$ indica que $T_1$, $T_3$ y $T_9$ son redundantes, se aprecia que utilizando las 10 transformaciones se obtiene un AUROC de 99.18±0.11 y tras la selección este disminuye a 98.94±0.18, es decir la invarianza no es perfecta y en este caso existen sinergias que al eliminar $T_1$, $T_3$ o $T_9$ hacen que disminuya el desempeño de GeoTransform.

El método de selección $M_{2.1.a}$ fue estudiado en detalle en la publicación [4] del tesista, donde es aplicado sobre GeoTransform72 y GeoTransform99 entrenando en small-HiTS y

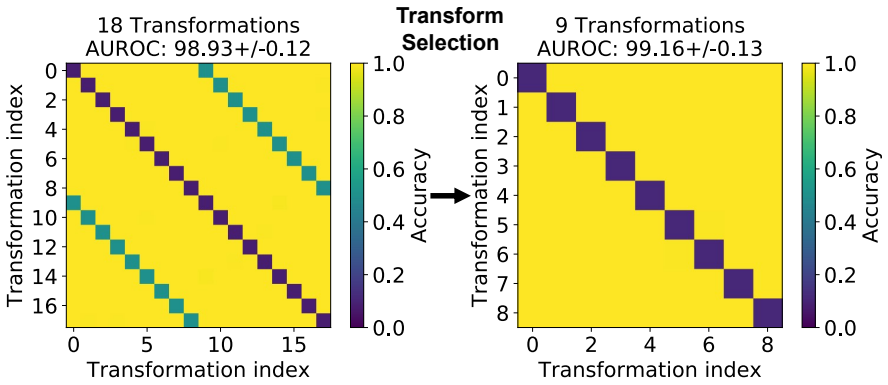



Figura 4.10: Matriz de discriminación para los *inliers* de la base de datos small-HiTS, usando un conjunto de transformaciones compuesto por 18 transformaciones; *flip* vertical y traslaciones. De forma similar a una matriz de correlación, el triángulo superior e inferior de las matrices son idénticos y las diagonales no significan nada. La matriz muestra que las traslaciones son fáciles de discriminar entre si ($\sim$100 % *accuracy*), excepto por los pares de transformaciones $(T_{\mathrm{i}}, T_{\mathrm{i+9}}), \mathrm{i} \in \{1, 2, ..., 9\}$ (cuadrados verdes) que obtienen un *accuracy* de $\sim$50 %. Un clasificador es incapaz de distinguir entre una colección de imágenes trasladadas y otra colección con la misma traslación, pero que se le ha aplicado a las imágenes un *flip* vertical de forma previa. En el lado derecho se muestra la matriz de discriminación tras eliminar transformaciones redundantes y consecuentemente la selección de transformaciones *útiles*. En este ejemplo, el AUROC mejora tras seleccionar las 9 transformaciones desde las 18 originales (eliminando la transformación de *flip* vertical). Figura obtenida de [4].

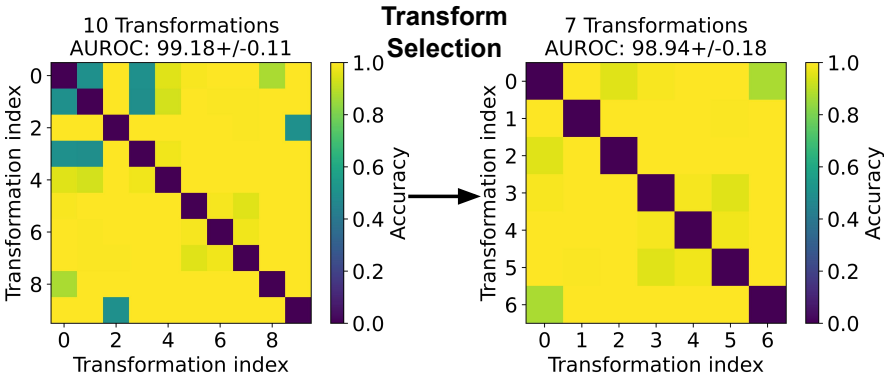



Figura 4.11: Selección de transformaciones con método $M_{2.1.a}$ sobre conjunto de 10 transformaciones de juguete $\mathcal{T}_{juguete}$ en base de datos small-HiTS. A la izquierda la matriz de discriminación para las 10 transformaciones, donde se identifica como redundantes $T_1$, $T_3$ y $T_9$. A la derecha se observa la matriz de discriminación con las 7 transformaciones que quedan tras descartar las redundantes según $M_{2.1.a}$. Se aprecia una disminución en AUROC tras la selección de transformaciones.

small-ALeRCE-ZTFv1. La tabla 4.6, resume los resultados obtenidos. En esta tabla, además del AUROC, se calcula el *accuracy* de todos los modelos.

Al aplicar el método de selección $M_{2.1.a}$ a GeoTransform72, en ambas bases de datos sólo se seleccionan las 9 transformaciones de traslación, con GeoTransform9. Este método de se-

Tabla 4.6: Desempeño de método $M_{2.1.a}$ sobre GeoTransform72 y GeoTransform99. GeoTransform72 se reduce a GeoTransform9 para ambas bases de datos. En el caso de small-ALeRCE-ZTFv1, GeoTransform9 obtiene un AUROC más bajo, pero de mayor *accuracy* que los que se logran con GeoTransform72. Aplicar el método $M_{2.1.a}$ sobre GeoTransform99 se reduce el número de transformaciones a GeoTransform35 para small-HiTS y GeoTransform29 para small-ALeRCE-ZTFv1. Los últimos dos modelos obtienen los mejores resultados en términos de AUROC.

| Modelo | Método de selección | small-HiTS | | small-ALeRCE-ZTFv1 | |
|---|---|---|---|---|---|
| | | AUROC | *Accuracy* | AUROC | *Accuracy* |
| GeoTransform72 | Ninguno | 98.78±0.26 | 96.97±0.55 | 87.33±1.24 | 77.80±0.94 |
| GeoTransform9 | $M_{2.1.a}$ sobre GeoTransform72 | 99.16±0.13 | 97.46±0.22 | 86.49±1.61 | 78.11±1.33 |
| GeoTransform99 | Ninguno | 99.12±0.04 | 97.59±0.19 | 90.80±0.61 | 83.53±0.66 |
| (small-HiTS) GeoTransform35 (small-ALeRCE-ZTFv1) GeoTransform29 | $M_{2.1.a}$ sobre GeoTransform99 | **99.20±0.06** | 97.21±0.10 | **91.39±0.76** | 82.81±0.36 |
| p-valor de t-test de Welch GeoTransform72 v/s GeoTransform35 & GeoTransform29 | | $6.8\times10^{-4}$ | $2.1\times10^{-1}$ | $5.3\times10^{-7}$ | $1.1\times10^{-8}$ |

lección permite reducir la dimensionalidad del problema sin sacrificar desempeño, a excepción de small-ALeRCE-ZTFv1, donde el AUROC disminuye en 0.84 puntos, sin embargo el *accuracy* aumenta en 0.31. Para GeoTransform99, se esperaba que sólo las 36 transformaciones de ambos kernels aplicados sobre las transformaciones fueran seleccionadas, pero en small-HiTS 35 transformaciones fueron seleccionadas (GeoTransform35), dejando fuera la transformación de kernel Gaussiano sin traslación, y en small-ALeRCE-ZTFv1 se seleccionaron 29 transformaciones (GeoTransform29), descartando todas excepto 2 de las transformaciones sólo con kernel Gaussiano. La Tabla 4.6 muestra que GeoTransform35 para small-HiTS y GeoTransform29 para small-ALeRCE-ZTFv1 son mejores que GeoTransfom99, i.e., existe una mejora en el AUROC cuando se aplica el método de selección $M_{2.1.a}$ sobre GeoTransform99 para ambas bases de datos. De acuerdo al test de hipótesis de Welch, hay diferencias que son estadísticamente significativas entre GeoTranform72 y los mejores modelos de la Tabla 4.6, para ambas bases de datos.

#### 4.2.3.3. Método de selección 3 - Selección por desempeño progresivo y regresivo

El método $M_3$ de variantes progresiva y regresiva, busca descartar transformaciones del tipo $T_{inutil.3}$, aquellas que al ser agregadas al conjunto de transformaciones, disminuyen el desempeño final de GeoTransform. Para evitar hacer trampa y mirar anomalías en el proceso de selección, es que se usa otra base de datos como *anomalías proxy*, cuando se entrena con la base de datos small-HiTS se utilizan anomalías de small-ALeRCE-ZTF-v1 como anomalías proxy, y vice-versa.

La variante $M_3$-progresiva funciona partiendo con el conjunto de datos original, sin transformaciones (equivalente a la transformación identidad $T_0$) y luego se agregan de a una las transformaciones del conjunto de transformaciones a evaluar, manteniendo en el conjunto se-

leccionada aquella que obtiene el mayor AUROC-proxy, este proceso se repite iterativamente hasta que el conjunto de transformaciones esté compuesto por todas las transformaciones disponibles. Al finalizar el proceso se escoge como mejor conjunto de transformaciones $\mathcal{T}_f$ aquel conjunto que posee el AUROC-proxy más alto de todas las iteraciones.

La variante $M_3$-regresiva comienza con el conjunto completo de transformaciones $\mathcal{T}_\text{i}$ y prueba sacando una transformación a la vez, dejando fuera aquella que obtiene el menor AUROC-proxy, este proceso se repite iterativamente hasta que el conjunto de transformaciones queda reducido a $T_0$. Finalizado el proceso completo, se selecciona el conjunto de transformaciones con mayor AUROC-proxy de todas las iteraciones. Tanto $M_3$-progresivo como $M_3$-regresivo se utilizan para evitar evaluar todas las posibles combinaciones de transformaciones. Para ambas variantes se necesitan entrenar muchos clasificadores, por lo cual en la variante progresiva solo se aceptan conjuntos iniciales con menos de 30 transformaciones. Para la variante regresiva (aún más costosa) se aceptan conjuntos iniciales menos de 20 transformaciones. Estas restricciones se basan en que cada variante se demora 7 días en completar el proceso de selección al utilizar los topes estipulados.

Los resultados de $M_3$-progresivo y $M_3$-regresivo para small-HiTS y el conjunto de transformaciones $\mathcal{T}_{juguete}$ pueden verse en la Tabla 4.7. En esta tabla se muestra que para ambos métodos disminuye el AUROC promedio obtenido por GeoTransform, 0.04 puntos de disminución para $M_3$-progresivo (diferencia no estadísticamente significativa con un p-valor de 0.54) y 0.14 para $M_3$-regresivo (diferencia estadísticamente significativa con un p-valor de 0.02). Además, en ambos casos se logra descartar la transformación $T_6$ (re-ordenamiento aleatorio de pixeles) que se previó de tipo $T_{inutil.3}$ debido a que no agrega información extra que sea útil para discriminar bogus. Para $M_3$-progresivo se descartan las transformaciones $T_1$, $T_7$, $T_8$ y $T_9$ (*flip*, ruido gaussiano, multiplicación por 0 y flip vertical con traslación). En el caso de $M_3$-regresivo, se eliminan las transformaciones $T_1$, $T_2$ y $T_3$ (*flip*, traslación, rotación). Para $M_3$-regresivo llama la atención que no se borra la transformación multiplicación por 0 ($T_8$), lo que hace intuir que este método necesita más revisión para determinar si es que presenta consistencia y congruencia en su utilidad para seleccionar transformaciones.

Tabla 4.7: AUROC de GeoTransform tras seleccionar transformaciones con $M_3$-progresivo y $M_3$-regresivo sobre small-HiTS y el conjunto de 10 transformaciones $\mathcal{T}_{juguete}$. El p-valor mostrado se calcula entre el AUROC original y el AUROC tras selección.

| Método de selección | # transformaciones seleccionadas (transformaciones eliminadas) | AUROC original | AUROC tras selección | p-valor de t-test de Welch |
|---|---|---|---|---|
| $M_3$-progresivo | 6 ($T_1$, $T_7$, $T_8$, $T_9$) | 99.18±0.11 | 99.14±0.17 | 0.54 |
| $M_3$-regresivo | 7 ($T_1$, $T_2$, $T_3$) | 99.18±0.11 | 99.04±0.14 | 0.02 |

#### 4.2.3.4. Flujos de selección

Todos los métodos de selección de transformaciones presentados anteriormente pueden ser mezclados de múltiples maneras para construir complejos flujos de selección de transformaciones. A continuación se presentan resultados de flujos donde se mezclan diferentes métodos.

La forma de mezclar métodos de selección obedece a un orden secuencial desde métodos de tipo $M_1$, a $M_2$ y luego $M_3$. Esto ya que $M_1$ descarta transformaciones triviales que han de ser eliminadas lo antes posible. $M_2$ elimina todo tipo de transformaciones redundantes, para dar paso a métodos de tipo $M_3$, los más costosos en términos de tiempo de ejecución y por lo cual han de recibir un conjunto de transformaciones inicial con pocas transformaciones. Además, los métodos tipo $M_3$ presentan ciertas inestabilidades y falta de justificación teórica en la forma en que descartan transformaciones. Al mezclar dos o más métodos en un flujo se utiliza el simbolo de composición $\circ$ para nombrar el flujo, por ejemplo, $M_1 \circ M_{2.1.a}$ es un flujo donde primero se aplica $M_1$ y luego $M_{2.1.a}$.

La evaluación de flujos de selección es un experimento que requiere un exploración más profunda y acabada, sirviendo exclusivamente como un ejercicio exploratorio y preliminar.

Los resultados se muestran en términos de AUROC promedio de 10 instancias de entrenamiento de GeoTransform sobre las bases de datos small-HiTS y small-ALeRCE-ZTFv1, utilizando los conjuntos de transformaciones de juguete $\mathcal{T}_{juguete}$, el conjunto utilizado para GeoTransform72 ($\mathcal{T}_{72}$) y GeoTransform99 ($\mathcal{T}_{99}$), este último incluye transformaciones de kernel sobre traslaciones. Los resultados para la base de datos small-HiTS se muestran en la Tabla 4.8, mientras que para small-ALeRCE-ZTFv1 se muestran en la Tabla 4.9. En ambas tablas, el mejor desempeño de cada columna está resaltado con texto en negrita, mientras que un resultado "*No calculado*", como los que se observan al utilizar $M_3$, indica que el conjunto de transformaciones que entra al método es muy grande cómo para ser evaluado en un tiempo prudente (entiéndase como prudente menos de 7 días de cómputo).

Tabla 4.8: AUROC para diferentes flujos de selección de transformaciones sobre base de datos small-HiTS, en conjuntos de transformaciones $\mathcal{T}_{juguete}$, GeoTransform72 y GeoTransform99. En negrita se indica el mejor resultado de cada columna. Un *"No calculado"* significa que alguno de los métodos $M_3$ recibe más transformaciones que los límites fijados en este trabajo.

| Flujo de selección de transformaciones | 10 transformaciones con $\mathcal{T}_{juguete}$ AUROC (# transformaciones seleccionadas) | GeoTransform72 AUROC (# transformaciones seleccionadas) | GeoTransform99 AUROC (# transformaciones seleccionadas) |
|---|---|---|---|
| Ninguno | 99.18±0.11 (10) | 98.78±0.26 (72) | 99.12±0.04 (99) |
| $M_1$ | **99.18±0.10 (9)** | 98.78±0.26 (72) | 99.12±0.04 (99) |
| $M_{2.1.a}$ | 98.94±0.18 (7) | **99.16±0.13 (9)** | 99.20±0.06 (35) |
| $M_1 \circ M_{2.1.a}$ | 99.07±0.16 (6) | 99.16±0.13 (9) | 99.20±0.06 (35) |
| $M_1 \circ M_{2.1.a} \circ$ $M_3$-progresivo | 99.07±0.16 (6) | 98.53±0.17 (4) | **99.42±0.09 (21)** |
| $M_1 \circ M_{2.1.a} \circ$ $M_3$-regresivo | 99.07±0.16 (6) | No calculado | No calculado |
| $M_3$-progresivo | 99.14±0.17 (6) | No calculado | No calculado |
| $M_3$-regresivo | 99.04±0.14 (7) | No calculado | No calculado |

Al analizar el AUROC obtenido por las estrategias de selección sobre el conjunto de 10 $\mathcal{T}_{juguete}$, se aprecia que para ambas bases de datos, en la mayoría de los flujos de selección

Tabla 4.9: AUROC para diferentes flujos de selección de transformaciones sobre base de datos small-ALeRCE-ZTFv1, en conjuntos de transformaciones $\mathcal{T}_{juguete}$, GeoTransform72 y GeoTransform99. En negrita se indica el mejor resultado de cada columna. Un *"No calculado"* significa que alguno de los métodos $M_3$ recibe más transformaciones que los límites fijados en este trabajo.

| Flujo de selección de transformaciones | 10 transformaciones con $\mathcal{T}_{juguete}$ AUROC (# transformaciones seleccionadas) | GeoTransform72 AUROC (# transformaciones seleccionadas) | GeoTransform99 AUROC (# transformaciones seleccionadas) |
|---|---|---|---|
| Ninguno | 91.97±0.77 (10) | 87.33±1.24 (72) | 90.80±0.61 (99) |
| $M_1$ | **92.13±1.69 (9)** | 87.33±1.24 (72) | 90.80±0.61 (99) |
| $M_{2.1.a}$ | 90.99±1.15 (7) | 86.49±1.61 (9) | 91.39±0.76 (29) |
| $M_1 \circ M_{2.1.a}$ | 90.56±1.59 (6) | 86.49±1.61 (9) | 91.39±0.76 (29) |
| $M_1 \circ M_{2.1.a} \circ$ $M_3$-progresivo | 76.27±5.85 (4) | **89.10±2.96** (6) | **94.14±0.39 (17)** |
| $M_1 \circ M_{2.1.a} \circ$ $M_3$-regresivo | 76.27±5.85 (4) | No calculado | No calculado |
| $M_3$-progresivo | 87.23±3.35 (7) | No calculado | No calculado |
| $M_3$-regresivo | 88.43±1.95 (8) | No calculado | No calculado |

el desempeño disminuye respecto a no aplicar ningún flujo de selección, excepto para small-ALeRCE-ZTFv1, donde sólo usar $M_1$ mejora los resultados. Que disminuya el AUROC en estos casos no resulta de especial relevancia, ya que este conjunto de 10 transformaciones sólo se usa para validar la capacidad de los flujos para descartar transformaciones de tipo $T_{inutil}$, siendo resultados experimentales preliminares que requieren estudios en profundidad, además de que muchos de ellos reportan resultados que no tienen diferencias estadísticamente significativas respecto a no utilizar flujos de selección. Se ha de destacar que todos los flujos probados son capaces de descartar al menos una transformación. Otro resultado que resalta en los experimentos de flujos de selección es que seleccionar transformaciones sobre $\mathcal{T}_{juguete}$ obtiene resultados similares a seleccionar transformaciones sobre $\mathcal{T}_{72}$, tanto en AUROC promedio como en desviación estándar, particularmente para small-ALeRCE-ZTFv1 flujos de selección como $M_1$ superan con creces todos los resultados obtenidos sobre $\mathcal{T}_{72}$ (excepto para el flujo de selección $M_1 \circ M_{2.1.a} \circ M_3$-regresivo, donde no hay diferencias estadísticamente significativas con $M_1$). Esto habla de la importancia de contar con transformaciones de calidad más que una gran cantidad de transformaciones.

Para GeoTranform72 sobre small-HiTS, en la Tabla 4.8 se observa que el mejor resultado es utilizar $M_{2.1.a}$ con un AUROC de 99.16±0.13 y 9 transformaciones seleccionadas (sólo traslaciones). Al aplicar el flujo $M_1 \circ M_{2.1.a} \circ M_3$-progresivo sobre small-HiTS, $M_1$ no borra transformaciones, $M_{2.1.a}$ deja las mismas 9 transformaciones mencionadas anteriormente y $M_3$-progresivo borra 5 transformaciones más, dejando sólo 4 transformaciones, lo que genera una reducción considerable en AUROC hasta 98.53±0.17, pese a esta abrupta disminución de desempeño se obtiene un AUROC no menor considerando que sólo se usan 4 transformaciones.

Sobre la base de datos small-ALeRCE-ZTFv1 con GeoTranform 72, en la Tabla 4.9, el método $M_{2.1.a}$ selecciona las mismas transformaciones que para small-HiTS. Para small-ALeRCE-ZTFv1 el mejor desempeño se obtiene con el flujo $M_1 \circ M_{2.1.a} \circ M_3$-progresivo, el que descarta 3 transformaciones de traslación dejando finalmente sólo 6 transformaciones que obtienen un AUROC de 89.10±2.96.

En el caso de GeoTranform99, tanto en la Tabla 4.8 de small-HiTS como en la Tabla 4.9 de small-ALeRCE-ZTFv1 se observa que el mejor resultado es utilizar el flujo que incluye $M_1 \circ M_{2.1.a} \circ M_3$-progresivo, con AUROC de 99.42±0.09 y 21 transformaciones seleccionadas para small-HiTS, y 94.14±0.39 y 17 transformaciones seleccionadas para small-ALeRCE-ZTFv1. Estos resultados mejoran los resultados del estado del arte reportados en [4]. En este caso no hay transformaciones triviales, por lo cual $M_1$ no elimina transformaciones. Si se analiza el método $M_{2.1.a}$ para small-HiTS, este elimina rotaciones, *flips* y utilización de kernel gaussiano sobre $T_0$. Mientras que para small-ALeRCE-ZTFv1 $M_{2.1.a}$ eliminan las mismas transformaciones que para HiTS, pero también se descartan gaussianas sobre traslaciones.

### 4.2.4. Comparación con trabajos relacionados

La Tabla 4.10 muestra una comparación entre diferentes métodos de detección de anomalías clásicos y basados en aprendizaje profundo, entre los cuales se incluye GeoTransform y sus versiones mejoradas sobre las bases de datos small-HiTS y small-ALeRCE-ZTFv1. Para ambas bases de datos, todas las versiones mejoradas de GeoTransform obtienen un mejor desempeño que la versión original de GeoTransform72 (la que incluye un esquema de entrenamiento sin detención temprana). Al aplicar el flujo de selección de transformaciones $M_1 \circ M_{2.1.a} \circ M_3$-progresivo sobre GeoTransform99 genera los modelos con el mejor desempeño: GeoTransform21 para small-HiTS y GeoTransform17 para small-ALeRCE-ZTFv1. El desempeño de estos modelos mejorados de GeoTransform tiene diferencias estadísticamente significativas con el desempeño de GeoTransform72 original, de acuerdo a un test de hipótesis de Welch.

## 4.3. Detección automática de bogus para la clasificación de objetos astronómicos

En la búsqueda de utilizar GeoTransform para detectar bogus sin la necesidad de intervención humana, y utilizar estos bogus para entrenar modelos de clasificación como el stamp classifier, es que se comprueba la efectividad de GeoTransform para esta tarea, al evaluar el desempeño del stamp classifier sin metadatos al ser entrenado sobre ALeRCE-ZTFv7, pero con bogus provenientes de 3 fuentes diferentes:

- Bogus de GeoTransform: Etiquetados automáticamente por GeoTransform.
- Bogus tempranos: Simulan la etapa temprana de un estudio astronómico. 1907 bogus etiquetados por expertos.
- Bogus avanzados: Simulan un estudio astronómico en etapas avanzadas. 16,870 bogus etiquetados por expertos.

Debido a la falta de herramientas para determinar la mejor versión de GeoTransform, es

Tabla 4.10: Resultados de diferentes detectores de anomalías y las versiones mejoradas de GeoTransform sobre las bases de datos small-HiTS y small-ALeRCE-ZTFv1. El modelo con el nombre "original GeoTransform72" no incluye detención temprana.

| Model | small-HiTS AUROC | small-ALeRCE-ZTFv1 AUROC |
|---|---|---|
| RAW-OC-SVM [55] | 97.46±0.09 | 86.01±0.05 |
| CAE-OC-SVM [21] | 95.67±0.14 | 81.15±0.20 |
| IF [15] | 96.10±0.18 | 82.19±0.10 |
| DSEBM [60] | 93.52±1.66 | 77.69±0.01 |
| ADGAN [61] | 91.83±0.29 | 79.01±0.49 |
| MO-GAAL [62] | 85.69±0.18 | 74.07±0.67 |
| (•) Original GeoTransform72 [21] | 98.60±0.23 | 85.63±1.48 |
| GeoTransform72 | 98.78±0.26 | 87.33±1.24 |
| GeoTransform99 | 99.12±0.04 | 90.80±0.61 |
| (□) (small-HiTS) GeoTransform21 (■) (small-ALeRCE-ZTFv1) GeoTransform17 | **99.42±0.09** | **94.14±0.39** |
| p-valor de t-test de Welch (•) v/s (□) & (■) | $2.7\times10^{-7}$ | $5.5\times10^{-9}$ |

que se utilizan 3 modelos diferentes de GeoTransform para obtener bogus de GeoTransform: original GeoTransform72, GeoTransform99 y GeoTransform17 (obtenido de aplicar el flujo de selección $M_1 \circ M_{2.1.a} \circ M_3$-progresivo a GeoTransform99).

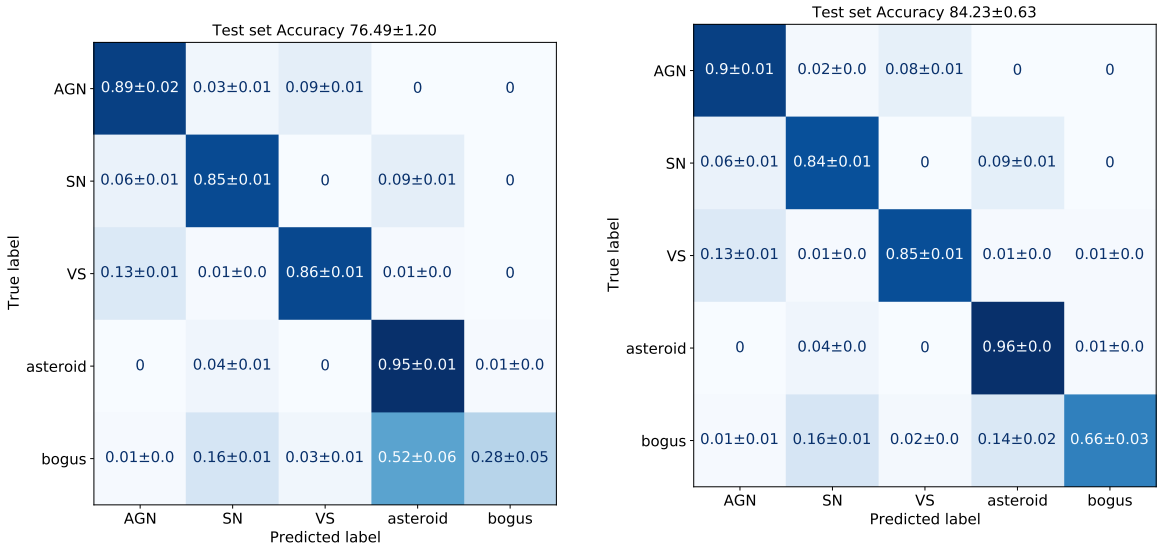
En la Tabla 4.11 se muestra el *accuracy* alcanzado al entrenar el stamp classifier sin metadatos con bogus de diferentes fuentes. Como era de esperarse, el *accuracy* más alto se obtiene al utilizar bogus avanzados, con 89.29±0.35. El utilizar bogus tempranos obtiene apenas un 76.49±1.20 de *accuracy*, el cual es superado con creces al utilizar bogus provenientes de GeoTransform. De las estrategias de usar bogus de GeoTransform, la que estadísticamente obtiene el mejor resultado es GeoTransform99, la que aumenta el *accuracy* de utilizar solo bogus tempranos en 7.74 puntos de *accuracy*. De la Tabla 4.11, se ve que GeoTransform17 obtiene el *accuracy* promedio más bajos de las 3 estrategias que usan bogus GeoTransform, con un resultado estadísticamente similar a usar bogus de GeoTransform72 (p-valor de t-test de Welch GeoTransform72 v/s GeoTransform17 de 0.53) y con un resultado estadísticamente peor que usar bogus de GeoTransform99 (p-valor de t-test de Welch GeoTransform99 v/s GeoTransform17 de 0.01).

Para tener una idea del desempeño obtenido por cada una de las 5 clases con que se entrena el stamp classifier sin metadatos (SN, VS, AGN, asteroide y bogus), en la Figura 4.12, se muestran las matrices de confusión promedio del stamp classifier sin metadatos sobre el conjunto de test de ALeRCE-ZTFv7, al ser entrenado con diferentes fuentes de bogus. Debido a que los resultados de utilizar bogus de las diferentes variantes de GeoTransform son parecidos (sólo variando el desempeño sobre los bogus correctamente etiquetados y los bogus incorrectamente etiquetados como asteroides), en la Figura 4.12 sólo se muestra la matriz de confusión promedio para bogus de GeoTransform99 y bogus tempranos. No se muestra

Tabla 4.11: Resultados el stamp classifier sin metadatos sobre el conjunto de test de base de datos ALeRCE-ZTFv7, donde los bogus que se utilizan provienen de diferentes orígenes, se utilizan 3 versiones de GeoTransform para etiquetar bogus. El modelo con el nombre "original GeoTransform72" no incluye detención temprana. Los resultados de cada modelo son el resultado de 5 instancias de entrenamiento.

| Tipo de bogus | *Accuracy* |
|---|---|
| bogus de Original GeoTransform72 | 82.89±1.10 |
| bogus de GeoTransform99 | 84.23±0.63 |
| bogus de GeoTransform17 | 82.47±0.94 |
| bogus temprano | 76.49±1.20 |
| bogus avanzados | **89.29±0.35** |
| p-valor de t-test de Welch GeoTransform72 v/s GeoTransform17 | 0.53 |
| p-valor de t-test de Welch GeoTransform99 v/s GeoTransform17 | 0.01 |

la matriz de confusión para bogus avanzados, ya que es muy parecida a la de la Figura 4.6 (obtenida del stamp classifier sin metadatos entrenado y evaluado sobre la base de datos ALeRCE-ZTFv5) y la comparación que se desea resaltar es entre bogus de GeoTransform y bogus tempranos.




(a) Utilización de bogus tempranos.

(b) Utilización de bogus de GeoTransform99.

Figura 4.12: Matrices de confusión promedio de evaluar el stamp classifier sin metadatos en el conjunto de test de ALeRCE-ZTFv7. El modelo es entrenado utilizando bogus de diferentes fuentes. (a) Entrenamiento con bogus de etapa temprana. (b) Entrenamiento con bogus de GeoTransform99.

De las matrices de confusión de la Figura 4.12, se aprecia que en la mayoría de las clases el desempeño es similar, excepto para la clase asteroide (mirar la columna *asteroid* de ambas matrices de confusión). Al entrenar con bogus de etapa temprana muchos de los bogus se confunden con asteroides, mientras que al utilizar sólo bogus de GeoTransform17 esta confusión disminuye, aumentándose el *recall* promedio de la clase asteroide de 0.28 al entrenar con bogus de etapa temprana a 0.66 al entrenar con bogus de GeoTransform99.

# Capítulo 5

# Discusión

Al igual que los capítulos anteriores, este se divide en 3 grandes secciones: clasificación temprana de objetos astronómicos, detección de anomalías tipo bogus y la combinación de ambas en detección automática de bogus para la clasificación de objetos astronómicos. Al final se agrega una sección de dificultades actuales y trabajo futuro.

## 5.1. Clasificación temprana de alertas astronómicas

En general, el stamp classifier se presenta como un modelo simple y con un desempeño particularmente bueno sobre primeras alertas del estudio astronómico ZTF, por lo que es uno de los modelos más importantes de la colaboración ALeRCE.

Una parte importante del diseño del stamp classifier fue el ajuste de sus hiperparámetros, siendo el tamaño de las tripletas de imágenes de las alertas el más polémico (originalmente de 63$times$63 pixeles), ya que utilizar las imágenes completas otorga mayor información la cual de no ser útil debería aprender a ser ignorada por el modelo. Pero también contar con imágenes más grandes aumenta la dimensionalidad del problema y se entra en el dilema de la *maldición de la dimensionalidad* [17], donde si aumentan las dimensiones del problema se necesitan cada vez más datos para resolverlo de manera adecuada. Se conjetura que los mejores resultados se obtienen al utilizar imágenes de $21\times21$ debido a esta situación, ya que la base de datos ALeRCE-ZTFv5 contiene una cantidad de alertas en el orden de decenas de miles, lo que no es suficiente como para entrenar el stamp classifier de forma adecuada con imágenes más grandes, dado que el número de parámetros aumenta $\sim$10 veces (de 441 a 3969). Lo que puede resultar particularmente importante para las clases menos representadas como SNe, que apenas contemplan 1620 alertas en la base de datos ALeRCE-ZTFv5. Se necesita un análisis más acabado del tamaño óptimo para las estampas a la entrada del clasificador que se está utilizando, ya que puede ser un factor de diseño importante en futuros estudios astronómicos basados en flujos de alertas. Este tipo de análisis puede ser realizado con rapidez, debido a que como el stamp classifier presenta una arquitectura convolucional simple, entrenar una instancia del modelo toma menos de 10 minutos (al ser entrenado en una GPU RTX2080-Ti) lo que permite realizar muchas pruebas y variaciones en sus hiperparámetros.

El *accuracy* de 0.941±0.004 del stamp classifier resalta como un desempeño muy bueno, considerando que solo se está utilizando la primera alerta de un objeto, cuando tradicionalmente se utilizan curvas de luz con varios puntos para detectar objetos astronómicos reales como AGNs, SNe y VSs. Al analizar la matriz de confusión de la Figura 4.2 resalta que la clase con desempeño más bajo (*recall* promedio de 0.87) es SN, se aprecia que la mayor confusión en las clasificaciones se produce con las clases asteroide y bogus, lo que puede ser corregido al incluir una segunda o tercera alerta del objeto. Cuando existe más de una alerta para un objeto, se puede descartar con mucha seguridad que el objeto estudiado sea un bogus o a un asteroide, ya que es poco probable que un artefacto tipo bogus o que un asteroide en movimiento se encuentren en el mismo lugar en 2 o más alertas consecutivas, es por esto que modelos basados en curvas de luz suelen no tener que preocuparse de este tipo de objetos. Artefactos tipo bogus pueden ocurrir en una misma locación cuando son producto de saturación de pixeles por una estrella muy brillante, pero estos casos son de escasa ocurrencia. Pese a esta fácil solución, se ha de recordar que la idea del stamp classifier es la identificación temprana de objetos con una única alerta, especialmente de SNe.

Como se mencionó en el párrafo anterior, algunas SNe del conjunto de test de ALeRCE-ZTFv5 son incorrectamente identificadas como bogus, lo que puede atribuirse a que algunas SNe tienen un valor de *point spread function* (PSF) muy pequeña. Esto hace que fuentes reales se vean concentradas en pocos pixeles, pudiendo producirse una confusión con bogus del tipo pixeles quemados o rayos cósmicos, los que usualmente se manifiestan en las imágenes como un único o unos pocos pixeles brillantes adyacentes.

Uno de los aspectos que permitieron mejorar el desempeño del stamp classifier de forma sustancial es la inclusión de metadatos en su entrada, con lo que se obtiene una mejora de 5.8 puntos de *accuracy*. Sin embargo, se ha de tener en consideración que al utilizar metadatos pueden incluirse sesgos estadísticos por la representatividad de cada valor de los diferentes metadatos (por ejemplo un metadato en específico puede presentar un valor muy común en la base de datos, pero no necesariamente ser representativo de las alertas producidas por el estudio astronómico). Estos sesgos sólo se hacen evidentes al poner el modelo en producción y validarlo sobre datos *"del mundo real"*. Para evidenciar si es que en el stamp classifier existen sesgos que perjudiquen al modelo, en el artículo [2] se hacen pruebas sobre alertas no etiquetadas provenientes del flujo de alertas de ZTF, y con ellas evidenciar cómo cada clase predicha por el stamp classifier se distribuye en el cielo observable, donde, por ejemplo SNe y AGNs deberían aparecer fuera del eje de la galaxia, al ser objetos que normalmente se ubican fuera de nuestra galaxia. Visualizaciones con detalles de cómo se distribuyen los valores de los metadatos de la base de datos ALeRCE-ZTFv5, pueden verse en [2].

Una preocupación que se tiene al diseñar la metodología de entrenamiento del stamp classifier, es la poca cantidad de alertas que se tiene de la clase SN, lo que hace necesario tener un conjunto de muestras de test y validación reducido, con el fin de mantenerlos balanceados. Pruebas preliminares realizadas con diferentes versiones de la base de de datos ALeRCE-ZTF muestran que utilizar diferentes semillas aleatorias para particionar los datos en diferentes conjuntos y aumentar el conjunto de muestras por clase en conjuntos de test y validación a 200 y 300 muestras por clase, generan desempeños similares en todos los casos. Por lo cual 100 muestras por clase en conjuntos de test y validación, es una elección adecuada para mantener la mayor cantidad de SNe en el conjunto de entrenamiento, en vista de que se tienen apenas

1620 SNe disponibles en la base de datos ALeRCE-ZTFv5.

La adición del regularizador de entropía es una de la innovaciones agregadas al stamp classifier, que si bien no mejora el desempeño del modelo en forma significativa, si ayuda a astrónomos a tomar decisiones más informadas para corroborar las predicciones del modelo sobre alertas nunca antes vistas. Lo que se hace especialmente útil cuando se usa el stamp classifier dentro del SN Hunter, herramienta que está en continuo uso y mejora por el equipo de ALeRCE, con el objetivo de ser implementada en estudios astronómicos futuros.

El stamp classifier es el primer modelo que intenta realizar clasificación sobre las 5 clases de objetos astronómicos definidas en este trabajo. Al ser el primero modelo de este tipo, se hace imposible la comparación de su desempeño con otros modelos de la literatura. Esta problemática se menciona en la publicación del stamp classifier [2], donde en forma alternativa a una comparación directa con otro modelo, se mide el grado de congruencia entre las clasificaciones realizadas por el stamp classifier y el LCC de ALeRCE [1] sobre un conjunto de alertas no etiquetadas, provenientes de ZTF. Con esto se busca ver si las clasificaciones del stamp classifier sobre objetos del tipo SN, AGN o VS son correctas, ya que el LCC se propone como un modelo más confiable para identificar entre estas clases, ya que realiza sus predicciones sobre curvas de luz con al menos 6 detecciones, lo que otorga más información de las dinámicas que caracterizan a cada objeto. Al realizar este experimento se encuentra que el stamp classifier clasifica como SN el 78 % de las alertas que el LCC predice que son SNe, 85 % de los AGNs y 96 % de las VS. Si es que se consideran las clasificaciones del LCC como etiquetas, entonces los porcentajes anteriores serían *recalls* por clase del stamp classifier. Información más detallada de estas comparaciones realizadas en [2] y [3], se encuentran en la Sección A.1 del Apéndice.

Si bien las comparaciones entre el stamp classifier y el LCC son un buen punto de partida, el LCC no puede ser aplicado sobre alertas de las clases asteroide o bogus, debido a que rara vez generan más de una detección. Al momento de la escritura de este trabajo se encuentran en desarrollo varias investigaciones por parte del equipo de ALeRCE para diseñar un modelo capaz de recibir más de una alerta en su entrada, uno de ellos recibe el nombre de *two-stamp-classifier* [5]. Este tipo de modelos resulta ser una simplificación del modelo propuesto en [30], donde se utilizan secuencias de imágenes de largo indefinido para clasificar objetos astronómicos. En [5] se compara el desempeño entre el two-stamp-classifier y el stamp classifier, sin regularización de entropía, sobre una versión de la base de datos ALeRCE-ZTF, y a diferencia del LCC, el two-stamp-classifier si es entrenado sobre bogus y asteroides, ya que una pequeña cantidad de estos tienen 2 detecciones, lo que hace que sea una comparación más justa entre ambos modelos. Al comparar ambos modelos el two-stamp-classifier obtiene un *accuracy* 93.7$\pm$0.3 %, mientras que el stamp classifier alcanza un *accuracy* de 92.3$\pm$0.5 % (mejora de 1.4 % puntos promedio), y lo que es más importante, el two-stamp-classifier obtiene un *recall* de 95$\pm$1 % % en la clase SN (mejora de 7 % puntos promedio de *recall*, respecto del stamp classifier), lo que supera con creces al stamp classifier y resulta muy prometedor para la detección de SNe. Las diferencias en términos de *accuracy* entre el two-stamp-classifier y el stamp classifier no son dramáticas, lo que confirma que el stamp classifier tiene un funcionamiento bastante bueno, pese a sólo utilizar la primera alerta de un objeto astronómico. Más detalles sobre las comparaciones realizadas en [5] entre el stamp classifier y el two-stamp-classifier pueden verse en la Sección A.1 del Apéndice.

## 5.2. Detección de anomalías tipo bogus

En lo que respecta a detección de anomalías, particularmente detección de bogus utilizando GeoTransform, se logró proponer mejoras al modelo las cuales se centran en mejorar los esquemas de entrenamiento, agregar nuevas transformaciones y seleccionar transformaciones. La mayoría del trabajo se centró en la selección empírica de transformaciones, la cual se denomina empírica ya que utilizan métodos inspirados en la intuición y que en la práctica mejoran el desempeño de GeoTransform, pero falta un desarrollo teórico que los fundamente y justifique.

Una de las principales dificultades que se encuentran al proponer mejoras al desempeño de GeoTransform y que es transversal a los algoritmos de detección de anomalías, es la falta de una estrategia de validación de los métodos sin utilizar el conjunto de test, lo que resulta difícil ya que para este tipo de modelos no se pueden utilizar anomalías durante su entrenamiento. Esto dificulta escoger cual de los modelos presentados es realmente el mejor y no se está sufriendo un sobre-ajuste al conjunto de test. Debido a que este es un problema transversal al problema de detección de anomalías, y que no es tratado normalmente en la literatura, donde comúnmente sólo se presentan resultados finales sobre un conjunto de test, se obvia en este trabajo y no se declara ningún modelo como el mejor, si no que se presentan resultados sobre una serie de modelos y se habla de aquel que *"presenta rendimiento superior"* en la base de datos utilizada. Si bien se proponen estrategias para mitigar estos efectos de sobre-ajuste y poder tomar decisiones más informadas de los algoritmos con mejor desempeño, como el utilizar 2 bases de datos (HiTS y ALeRCE-ZTF), estas estrategias no son del todo efectivas. Esta problemática aparece de forma transversal en todos los campos del aprendizaje de máquinas, donde muchas publicaciones sólo reportan sus resultados finales sobre el conjunto de test, sin reportar resultados en métricas de validación o no se menciona si es que se hicieron ajustes de hiperparámetros. En el afán de mejorar el estado del arte, se utilizan bases de datos que han sido utilizadas por muchos estudios, y si es que se obtienen mejores resultados estos se publican.

La primera mejora propuesta a GeoTransform se relaciona con evitar el sobre-ajuste al conjunto de entrenamiento, lo que se hace con una estrategia de detención temprana con *patience* 0, validando al final de cada época. Normalmente se usaría un *patience* más extenso que permita al modelo explorar el espacio de parámetros por más iteraciones y salir de mínimos locales en caso de tener un sobre-ajuste. Sin embargo se mantiene el *patience* 0, debido a que se obtienen resultados que mejoran y estabilizan el desempeño de GeoTransform, al aumentar su AUROC promedio y disminuir su varianza, sumado al hecho de que *patience* 0 permite entrenar modelos de forma rápida, lo que es conveniente para el volumen de experimentos que se realiza con GeoTransform. Otra razón para escoger *patience* 0 es que resulta ser una estrategia de detención temprana conservadora, al utilizar el conjunto de validación en pocas oportunidades, se evita tener un sobre-ajuste a este conjunto, el cual está compuesto únicamente por *inliers*, por lo que no refleja del todo la tarea final a resolver (AUROC en el conjunto de test). A fin de cuentas se prefiere la utilización de detención temprana para prevenir el sobre-ajuste de GeoTransform, ya que permite una holgura a la cantidad de épocas que necesita GeoTransform para ser entrenado en cada base de datos de forma adecuada (cada base de datos suele necesitar una cantidad de épocas de entrenamiento diferente). Por el contrario, entrenar por un número fijo de $200/\mathcal{T}$ iteraciones, como en el esquema original

de entrenamiento de GeoTransform, puede no funcionar bien para todas las bases de datos y entregar resultados inestables con altas desviaciones estándar.

En lo que respecta a la inclusión de nuevas transformaciones y selección de las mismas, se proponen una gran cantidad de métodos, de los cuales la mayoría demuestra aumentar el desempeño de GeoTransform en la detección de bogus. En primer lugar se desestiman declaraciones del artículo original de GeoTransform [21], donde se menciona que son necesarias transformaciones que preserven casi intacta la información de las imágenes originales. En esta tesis se mostró que al incluir transformaciones que incluyen operaciones de kernel, con intenciones de resaltar características conocidas a priori en bogus, se mejora el desempeño de GeoTransform. Esto se atribuye a que este tipo de transformaciones hace que *inliers* y bogus sean radicalmente diferentes y más fáciles de discriminar para GeoTransform.

Para la selección de transformaciones se sientan bases para un marco empírico de selección de transformaciones que incluye definiciones del tipo de transformaciones que se desea descartar ($T_{inutil}$) en un esquema de entrenamiento auto-supervisado como el de GeoTransform. El primer método que se propone para encontrar transformaciones $T_{inutil}$ es $M_1$, para descartar transformaciones que borran toda la información, en la práctica no se incluyen este tipo de transformaciones en conjuntos utilizados para entrenamiento auto-supervisado, ya que no hace sentido incluir una transformación que borre toda la información de las muestras originales, sin embargo se incluye este método por completitud. Un resultado interesante que se muestra al evaluar el método $M_1$ es que transformaciones que cambian las imágenes por ruido gaussiano (el valor de cada pixel de la imagen se muestrea de una distribución $\mathcal{N}(0,1)$) no son descartadas como triviales, al ser utilizadas sobre datos astronómicos, lo que puede deberse a que en bases de datos astronómicas el fondo de las imágenes contiene ruido asociado al cielo, que si bien presenta una distribución poisson, tiene semejanzas visuales al ruido Gaussiano, por lo cual existe algo de información en común entre la base de datos original y la que sólo contiene ruido gaussiano en las imágenes. Este fenómeno ocurre tanto en los *inliers* de la base de datos small-HiTS como de small-ALeRCE-ZTFv1. Este método de selección incluye un umbral $\varepsilon$ que debería ser ajustado para la base de datos que se está evaluando, pero el valor escogido de $\varepsilon = 10^{-2}$ funciona de forma exitosa para descartar transformaciones triviales tanto en small-HiTS como en small-ALeRCE-ZTFv1

En lo que respecta al método de selección $M_{2.1.a}$, donde se busca eliminar transformaciones redundantes, es que en el conjunto de transformaciones de juguete, tanto para small-HiTS como small-ALeRCE-ZTFv1, se logró eliminar transformaciones a las que las bases de datos son invariantes como rotaciones o *flips*, y transformaciones redundantes, con lo que se disminuye el desempeño de GeoTranform respecto a no eliminar ninguna transformación. Esto se atribuye a que las invarianzas o redundancias son imperfectas, es decir las imágenes generadas con las transformaciones no son completamente idénticas y aportan información que en cierta medida mejora el desempeño del algoritmo en la tarea final, es decir, existen sinergias en la información que estas transformaciones aparentemente redundantes aportan, por lo cual no pueden ser descartadas de manera tajante. Pese a esto, las disminuciones en desempeño son leves, siendo las sinergias algo que no se trata en este trabajo y se deja como una investigación propuesta para el futuro.

En cuanto a los métodos de selección $M_3$-progresivo y $M_3$-regresivo, estos presentan de-

sempeños y selección de características similares para el conjunto de transformaciones $\mathcal{T}_{juguete}$ sobre small-HiTS. Llama la incapacidad de $M_3$-regresivo descartar la transformación trivial de multiplicar las imágenes por 0, lo que delata su carácter de estrategia exploratoria que requieren de mayor investigación. Esto se suma a la limitante de ambas variantes de no poder ser aplicadas sobre conjuntos demasiado grandes de transformaciones.

Al observar los resultados de los flujos de selección para small-HiTS en la Tabla 4.8 y small-ALeRCE-ZTFv1 en la Tabla 4.9, algo que llama fuertemente la atención es como al utilizar el conjunto de juguete $\mathcal{T}_{juguete}$ con 10 transformaciones, se obtienen resultados que superan a todas las estrategias de selección para GeoTransform72, y en algunos casos superan las estrategias de selección para GeoTransform99. Esto da indicios de que contar con pocas transformaciones pero de calidad da mejores resultados que contar con una gran cantidad transformaciones que pueden resultar redundantes entre sí, lo que apoya la necesidad de contar con buenas estrategias de selección de transformaciones. Otra conjetura que se puede hacer a partir de los resultados de estas tablas, es que cuando se tiene una pequeña cantidad de transformaciones, las transformaciones redundantes no se pueden eliminar así como así, ya que las sinergias pueden estar jugando un rol importante, esto se evidencia en que al borrar transformaciones como rotaciones o *flips* en el conjunto de transformaciones $\mathcal{T}_{juguete}$ (supuestamente redundantes para small-HiTS y small-ALeRCE-ZTFv1), el desempeño del algoritmo disminuye. A partir de estos resultados nace la interrogante para investigaciones futuras de si el efecto de eliminar transformaciones sinérgicas parece ser menos importante cuando existe un mayor número de transformaciones redundantes, ya que factores de variación pueden ser codificados en otras transformaciones.

Como ya se ha mencionado varias veces, todo el trabajo desarrollado entorno a la selección de transformaciones es experimental y necesita ser estudiado con una mayor profundidad, pero los resultados mostrados en esta investigación demuestran la importancia de contar con estrategias de selección de transformaciones. Aquí se presentan algunos métodos que en la mayoría de los casos mejoran el desempeño del algoritmo, como $M_1$ y $M_{2.1.a}$, los cuales además son lo suficientemente estables como para entregar resultados similares cada vez que se somete a una base de datos y conjunto de transformaciones a selección.

Si bien no es posible declarar que el modelo donde se aplica el método de selección $M_1 \circ M_{2.1.a} \circ M_3$-progresivo sobre GeoTransform99 es la mejor versión de GeoTransform (siendo los modelos GeoTransform21 para small-HiTS y GeoTransform17 para small-ALeRCE-ZTFv1), debido a que puede existir un sobre-ajuste al conjunto de test, el que se obtenga el desempeño más alto tanto en small-HiTS como en small-ALeRCE-ZTFv1 (resultados estadísticamente significativos respecto a los modelos que les siguen en desempeño) resulta ser un buen indicador de que es un buen modelo y método de selección de transformaciones. La calidad de estos modelos también se aprecia al compararlos con otros modelos de la literatura para detección de anomalías, donde GeoTransform21 y GeoTransfor17 superan al resto de los modelos e incluso al modelo original de GeoTransform por un amplio margen.

## 5.3. Detección automática de bogus para la clasificación de objetos astronómicos

Al utilizar GeoTransform como estrategia de etiquetado automático de bogus, se logra superar el desempeño de utilizar bogus etiquetados por expertos en etapas tempranas de un estudio astronómico. Si bien no se logra llegar al desempeño alcanzado por bogus etiquetados a mano en etapas tempranas de un estudio astronómico, este nunca fue el objetivo, ya que resulta ambicioso superar la inspección y conocimiento experto acumulado por tanto tiempo. Sin embargo, se logran resultados suficientemente buenos como para incluir este modelo en el procesamiento de futuros estudios astronómicos y relevar o ayudar a expertos en su tarea de etiquetado de bogus y enfocar su tiempo en otras tareas de mayor interés. Si bien se hace necesaria una estrategia para seleccionar cual de los modelos de GeoTransform es el mejor para etiquetar bogus, los resultados obtenidos al entrenar el stamp classifier sin metadatos en cada caso no varían fuertemente, siendo cualquiera de las estrategias mejor que utilizar bogus de etapa temprana.

De los resultados de la 4.11, se aprecia que usar bogus de GeoTransform17 tiene un desempeño similar a usar bogus de GeoTransform72 y peor a usar bogus de GeoTransform99, este comportamiento puede atribuirse a la presencia de sobre-ajuste en las transformaciones seleccionadas para GeoTransform17, las que además fueron seleccionadas utilizando la base de datos *small*-ALeRCE-ZTFv1 y los bogus de ALeRCE-ZTFv7 presentan una dificultad más grande, al ser más parecidos a *inliers*.

En la Figura 4.12 se observan las matrices de confusión que comparan los resultados obtenidos por el stamp classifier sin metadatos con bogus de etapa temprana contra usar bogus etiquetados por GeoTransform99, donde se aprecia que las ganancias en *accuracy* principalmente vienen de resolver confusiones entre bogus y asteroides. Esto puede no ser útil para detectar SNe, objetivo principal para el que se diseñó el stamp classifier en el estudio astronómico de ZTF, pero para otros estudios astronómicos que se enfocan en la identificación de asteroides, como el estudio astronómico ATLAS [29], esta mejora en el desempeño del algoritmo resulta de especial interés.

En general, se puede declarar que el enfoque de mezclar el trabajo realizado para clasificación de objetos astronómicos con el stamp classifier y de detección de bogus con GeoTransform es una estrategia exitosa que permite utilizar GeoTransform para mejorar el stamp classifier al etiquetar bogus de forma automática en etapas tempranas de un estudio astronómico. Se ha de tener en cuenta que para que esta estrategia pueda ser utilizada se necesita de *inliers* etiquetados por *cross-match* así que no puede utilizarse de forma inmediata al comienzo de la operación de un estudio astronómico, se tiene que esperar una cierta cantidad de noches con el objetivo de que se genere una cantidad de alertas etiquetadas por *cross-match*, la que permita entrenar tanto el stamp classifier como GeoTransform.

## 5.4. Dificultades actuales y propuestas de trabajo futuro

### 5.4.1. Clasificación temprana de objetos astronómicos

Una de las principales dificultades actuales del stamp classifier es cómo mejorar su desempeño en SNe, clase en que presenta el peor desempeño de las 5 consideradas. La forma más evidente de incrementar el desempeño del modelo es recolectar más datos, particularmente de SNe, para mejorar su representatividad en el conjunto de entrenamiento, ya que es la clase con menor cantidad de muestras. Para esto el equipo de ALeRCE constantemente utiliza el SN Hunter para identificar SNe y reportarlas a TNS, las cuales una vez confirmadas son incorporadas en nuevas versiones de la base de datos, por ejemplo ALeRCE-ZTFv7. Otra forma de mejorar el desempeño, es mejorar la estrategia de balanceo de clases en el conjunto de entrenamiento, la que actualmente se realiza por simple replicación de datos, siendo que actualmente existen técnicas avanzadas de aprendizaje profundo para aumentar la cantidad de datos disponibles, entre estas estrategias resaltan las GANs, capaces de generar imágenes artificiales de apariencia realista, por lo cual se podría entrenar una GAN para generar imágenes de SNe o cualquier otra clase. Pese a lo prometedor de las GANs, su desventaja es que necesitan grandes cantidades de datos para ser entrenadas, lo que dificulta su utilización en generación de SNe. Otra dificultad de generar datos artificiales es que se han de generar tanto las tripletas de imágenes como los vectores de metadatos, siendo difícil lograr una congruencia de ambos.

Una forma sencilla que potencialmente puede mejorar los resultados del stamp classifier es utilizar estrategias más sofisticadas para la búsqueda de hiperparámetros. En vez de utilizar una búsqueda aleatoria de hiperparámetros, pueden utilizarse librerías como *hyperopt* [123], la que implementa estrategias de optimización bayesiana para la búsqueda de hiperparámetros.

Otro posible camino para mejorar el stamp classifier, es utilizar más de una alerta como entrada al modelo. Como se mencionó en secciones anteriores, esto permite disminuir la posibilidad de que una alerta sea gatillada por un bogus o un asteroide, ya que es muy raro que este tipo de eventos ocurran más de una vez en una misma región del cielo. Como se menciona en la Sección 5.1 anterior, actualmente existen esfuerzos por parte de investigadores del equipo de ALeRCE para diseñar este tipo de modelos, uno de ellos recibe el nombre de *two-stamp-classifier* [5].

Aplicaciones extra del stamp classifier como sistema de clasificación temprana de alertas, sería enfocarlo en la detección de otro tipo de objetos más allá de SNe, como la rápida detección de variabilidad extrema en AGN o el seguimiento de objetos del Sistema Solar, por ejemplo detección de asteroides.

### 5.4.2. Detección de anomalías tipo bogus

El trabajo realizado en detección de anomalías es el que presenta la mayor cantidad de desafíos de esta tesis, ya que es una tarea de particular dificultad, al no tenerse acceso a anomalías durante el proceso de entrenamiento. A raíz de esto surge la problemática evidente de cómo validar los modelos para escoger el mejor dentro de una serie de mejoras propuestas, y evitar de esta forma el sobre-ajuste al conjunto de test de la base de datos utilizada. Estos problemas se pueden abordar de forma superficial utilizando estrategias de validación cruzada

para evitar sobre-ajuste al conjunto de test, o como se hace en esta tesis, utilizar múltiples bases de datos. Sin embargo, no son una solución definitiva y que de garantías de evitar un sobre-ajuste. Esta debería ser una avenida fundamental en la investigación de detección de anomalías.

En lo que respecta al algoritmo de GeoTransform, en los experimentos de proposición y selección de transformaciones se hace evidente que seleccionar las transformaciones utilizadas es fundamental para el desempeño del modelo. Sin embargo, los métodos propuestos en esta tesis fueron estrategias exploratorias para sentar las bases del problema y requieren más estudio y profundización para mejorar su estabilidad y conexión con bases teóricas. La primera área que se ha de explorar en un trabajo futuro es la profundización del desempeño del modelo al ser entrenado con esquemas de entrenamiento para grandes bases de datos, donde queda pendiente evaluar su desempeño sobre las diferentes versiones de ALeRCE-ZTF.

Otra avenida interesante de trabajo futuro es abordar la selección de transformaciones para GeoTransform desde el campo de teoría de la información, tarea que no es tan simple debido a la alta dimensionalidad que tiene el dominio de las imágenes, donde las imágenes no pueden sólo verse como grupos de pixeles aislados, sino como elementos que tienen información codificada en su estructura espacial. Además de seleccionar transformaciones, sería de especial interés la posibilidad de aprender nuevas transformaciones, no solo seleccionarlas, sino que creando un método que permita crearlas u optimizarlas. La dificultad de este enfoque radica en que para GeoTransform, y en general para los algoritmos basados en aprendizaje auto-supervisado, no se tiene acceso a la tarea final que se busca resolver, como es el caso de AUROC en la clase bogus para esta tesis. Problemática que también se evidenció en la proposición de métodos de selección de transformaciones en este trabajo, particularmente para los métodos de selección de tipo $M_3$. Lo más cercano a aprendizaje de transformaciones que se ha explorado en la literatura es AutoAugment [124], estrategia de aumentación de datos, donde no se aprenden transformaciones de la nada, si no que se seleccionan los hiperparámetros que componen las transformaciones, como ángulos de rotación o cantidad de pixeles de traslación, los que se seleccionan con métodos de aprendizaje reforzado [125].

Como trabajo futuro en la selección de transformaciones, también se hace crucial incluir sinergias entre transformaciones, para no llegar y descartar aquellas que resultas similares a otras transformaciones, pero de todos modos incluyen factores de variación beneficiosos para el algoritmo.

Otra idea de trabajo futuro con el objetivo de mejorar la arquitectura de GeoTransform es mezclar GeoTransform con la técnica de outlier exposure [73], de forma similar a cómo se hace en [74], pero incluyendo el ajuste de la distribución de Dirichlet para calcular el puntaje normalidad, lo que podría traer mejoras al desempeño del algoritmo.

### 5.4.3. Detección automática de bogus para la clasificación de objetos astronómicos

La primera propuesta de trabajo futuro es resolver el sobre-ajuste en las transformaciones seleccionadas para GeoTransform17, para que la selección de transformaciones sea útil para encontrar bogus para entrenar el stamp classifier. Una alternativa para solucionar este

problema es aplicar las estrategias de selección de transformaciones sobre la base de datos ALeRCE-ZTFv7 y repetir los experimentos de esta sección. Sin embargo, estas estrategias necesitan modificaciones para poder ser aplicadas sobre grandes cantidades de datos, una alternativa es aplicar las estrategias de selección sobre un subconjunto reducido de datos de ALeRCE-ZTFv7, pero esto hace que el problema del sobre-ajuste sea más difícil de resolver.

Otra avenida de trabajo futuro que se hace evidente es encontrar una forma de incorporar bogus confirmados por expertos, por ejemplo los bogus de etapa temprana, tanto en el stamp classifier como en el entrenamiento de GeoTransform. La forma más fácil de lograr esto sería simplemente mezclar los bogus de etapa temprana y los bogus de GeoTransform para entrenar el stamp classifier. Para incorporar los bogus de etapa temprana en el entrenamiento de GeoTransform, se propone implementar outlier exposure (OE), que significa incluir bogus durante el entrenamiento de GeoTransform y cada vez que se muestra un bogus de etapa temprana, forzar al modelo a que genere clasificaciones de alta entropía e incorrectas, de tal forma de que GeoTransform sepa que ha de otorgar puntajes de normalidad bajo a muestras que se parezcan a los bogus de etapa temprana. Esto haría que se cambie a un enfoque de detección de anomalías supervisado o débilmente supervisado.

Otra propuesta de trabajo futuro es la inclusión de los metadatos en el modelo de GeoTransform, permitiendo la comparación entre el stamp classifier entrenado con metadatos y el esquema que integra GeoTransform y el stamp classifier.

Como trabajo futuro de ingeniería se propone implementar un sistema que integre GeoTransform, el stamp classifier y el SN Hunter en un ciclo de mejora continua de los algoritmos, el cual sea capaz de operar en estudios astronómicos apenas se cuente con una base de datos de *inliers* con *cross-match*. La operación de este sistema sería tomar la base de datos de *inliers* para entrenar GeoTransform con OE incluido (GeoTransform+OE), después detectar bogus del flujo de alertas generada por el estudio astronómico, luego utilizar la base de datos de *inliers* y los bogus etiquetados por GeoTransform+OE par entrenar el stamp classifier, el que está integrado en el SN Hunter, para que después el SN Hunter sea utilizado por expertos para identificar bogus mal etiquetados e incluirlos en el entrenamiento de GeoTransform+OE y del stamp classifier en un ciclo de mejora continua de los modelos.

# Capítulo 6

# Conclusiones

Como todos los capítulos anteriores, este también se divide en 3 secciones principales: clasificación temprana de objetos astronómicos, detección de anomalías tipo bogus y la combinación de ambos en la detección automática de bogus para la clasificación de objetos astronómicos. En cada una de estas secciones se indica si se cumplieron las hipótesis y objetivos del trabajo, sumado a un resumen del trabajo realizado y sus principales resultados. Se agrega una sección de resumen de contribuciones y últimos comentarios.

## 6.1. Clasificación temprana de objetos astronómicos

Se logró diseñar un clasificador llamado stamp classifier, el cual es implementado en el flujo de procesamiento de datos en línea del *broker* astronómico ALeRCE. Utilizando información de imágenes y metadatos contenidos en alertas del flujo de datos de ZTF se logró entrenar el stamp classifier para discriminar entre alertas de SN, AGN, VS, asteroides y bogus, utilizando sólo la primera alerta detectada. El stamp classifier hace uso de una CNN para resolver este problema de clasificación de forma automática y rápida, lo que permite identificar los mejores candidatos para realizar un posterior seguimiento con telescopios de precisión. Las entradas al stamp classifier son las imágenes de ciencia, referencia y diferencia, junto a un vector de 23 metadatos seleccionados de las alertas de primeras detecciones. La arquitectura convolucional del stamp classifier es invariante a rotaciones dentro de las imágenes y es entrenada con una función de costo con regularización de entropía, lo que permite mejorar la interpretabilidad de las probabilidades predichas para cada muestra, en términos del valor de certeza asignado a cada clase. De esta manera los expertos pueden comprender mejor la naturaleza de la alerta inspeccionada.

Para las 5 clases que identifica el stamp classifier se logra un *accuracy* de $0.941 \pm 0.004$ en el conjunto de test balanceado de la base de datos ALeRCE-ZTFv5. Este valor de *accuracy* es suficiente para declarar que se cumple la hipótesis de que en una alerta de la primera detección de objetos astronómicos existe información suficiente para discriminar entre las 5 clases de VS, AGN, SN, asteroide y bogus, siendo un modelo tan útil en la práctica que ha sido puesto en producción por el equipo de ALeRCE y es utilizado día a día por astrónomos. De este modo también se logra el objetivo O1 de implementar el stamp classifier.

Las predicciones del stamp classifier puesto en producción por ALeRCE se encuentran disponibles en la herramienta creada para la inspección de candidatos a SN, llamada SN Hunter (`https://snhunter.alerce.online/`). Esta herramienta permite la fácil visualización y análisis de SN por parte de expertos, con el objetivo de reportarlas para seguimiento a TNS. Este proceso de inspección manual ha permitido que entre el 26 de Junio del 2019 y el 28 de Febrero del 2021 el equipo de ALeRCE haya reportado más de 6846 nuevos candidatos a SN a TNS, de los cuales 995 han sido observados por telescopios de precisión y 971 han sido confirmados como SN.

## 6.2. Detección de anomalías tipo bogus

Se propuso una serie de mejoras al algoritmo GeoTransform para la detección de bogus en dos bases de datos astronómicas (small-HiTS y small-ALeRCE-ZTFv1). En primer lugar, se proponen estrategias de entrenamiento basadas en detención temprana para evitar el sobre-ajuste del clasificador auto-supervisado de GeoTransform. También, se introducen nuevas transformaciones basadas en filtros Laplaciano y Gaussiano para resaltar propiedades conocidas de bogus y hacerlos más fáciles de distinguir frente a imágenes de *inliers* (objetos astronómicos reales). Además, se propone una metodología empírica para descartar transformaciones, la cual sienta las bases y la importancia de la selección del conjunto adecuado de transformaciones para GeoTransform según la base de datos que se está usando. A partir de esta metodología se proponen tres métodos para seleccionar transformaciones, uno basado en teoría de la información que elimina transformaciones que borran toda la información original de las imágenes, dos métodos que buscan descartar transformaciones redundantes o transformaciones a las que los *inliers* de las bases de datos resultan invariantes, y otros dos métodos para descartar transformaciones que disminuyen el desempeño de GeoTransform, los que se basan en estrategias de selección de características.

El esquema de entrenamiento propuesto para GeoTransform, basado en detención temprana permite no solo mejorar el desempeño del algoritmo sino que ayuda obtener un entrenamiento más estable y confiable, no asignando un número arbitrario de iteraciones que puede producir un sobre-ajuste no deseado al conjunto de entrenamiento.

En la Tabla 4.5 se aprecia cómo la introducción de nuevas transformaciones basadas en kernel mejora los resultados de utilizar GeoTransform72 en todos los casos probados, tanto para la base de datos small-HiTS como small-ALeRCE-ZTFv1. El mejor desempeño con nuevas transformaciones se logra con GeoTransform99 que obtiene un AUROC de 99.12 en small-HiTS y de 90.8 en small-ALeRCE-ZTFv1. Este comportamiento permite afirmar con cierto grado de seguridad que se cumple la hipótesis que estipula que utilizar transformaciones que resalten características particulares de bogus permite mejorar el desempeño de GeoTransform. Esto también permite concluir que se logra parte del objetivo O2 de superar el AUROC al proponer nuevas transformaciones.

Una de las líneas principales de esta investigación se centró en la selección de transformaciones adecuadas para la estrategia de aprendizaje auto-supervisado de GeoTransform, arista de investigación de la cual no se han encontrado registros previos hasta la fecha. Se sientan bases y definiciones basadas en intuiciones empíricas, definiendo tres posibles fuentes de transformaciones inútiles que se desean descartar, además de métodos para encontrarlas.

Al realizar esto se satisface el objetivo O3 de esta tesis. Debido a lo novedoso y desafiante de esta aproximación a la selección de transformaciones, se aborda el problema de forma exploratoria y experimental, mostrando resultados sobre los cuales no se busca tomar decisiones definitivas sino que sólo resultan útiles como estudio preliminar. Los experimentos del conjunto de transformaciones de juguete $\mathcal{T}_{juguete}$ para evidenciar la capacidad de los métodos de selección de transformaciones obtuvieron resultados satisfactorios, eliminando las transformaciones que se esperaba fuesen inútiles, donde se aprecia que la base de datos small-HiTS resulta parcialmente invariante a rotaciones y *flips* según el método $M_{2.1.a}$. De los experimentos con este conjunto de transformaciones, se muestra que al borrar transformaciones aparentemente inútiles, el desempeño de GeoTransform disminuye, lo que da cuenta de lo importante que es incluir sinergias para realizar una selección adecuada de transformaciones, por lo cual se hace énfasis en la necesidad de investigar esta propiedad en un trabajo futuro.

Utilizando los métodos propuestos para selección de transformaciones, se logra reducir GeoTransform72 a GeoTransform9 en ambas bases de datos, GeoTransform99 se reduce a GeoTransform21 para small-HiTS y GeoTransform17 para small-ALeRCE-TFv1. Además de reducir la dimensionalidad del problema y tiempo de ejecución de GeoTransform, con estos métodos se obtiene una mejora considerable en el AUROC de GeoTransform, siendo los modelos con el AUROC más alto en ambas bases de datos, con lo que se supera el desempeño del estado del arte reportado en [4]. Debido a que con cada método utilizado se obtienen diferentes conjuntos de transformaciones para cada base de datos, y que el desempeño es mejorado con creces en cada una, puede responderse con cierto grado de seguridad que se cumple la hipótesis de que el mejor conjunto de transformaciones para GeoTransform depende de la base de datos utilizada, y que existen métodos capaces de encontrar mejores conjuntos para descartar transformaciones que afecten el desempeño de GeoTransform. Si bien los métodos de esta tesis no aseguran encontrar *"el mejor"* conjunto de transformaciones, son un paso en la dirección adecuada para encontrar dicho conjunto. Se requieren más estudios y experimentos para refinar los métodos de selección propuestos, ya que estos pueden presentar resultados inestables y algunos dependen de umbrales que podrían ser fijados de mejor manera (como $M_1$). El método que demuestra mejor consistencia sobre cada base de datos es $M_{2.1.a}$, ya que se descartan las transformaciones esperadas en $\mathcal{T}_{juguete}$ y siempre que se ejecuta se descartan las mismas transformaciones, pese a contar con el componente estocástico de depender de un clasificador. Pese a que la utilización del método $M_3$ está incluida en el flujo de selección que obtiene mejores resultados en las bases de datos reducidas, su uso debe ser investigado en detalle, ya que su funcionamiento incluye la utilización de una base de datos externa que aporte con anomalías-proxy, lo que lo hace altamente inestable y normalmente obtiene resultados diferentes cada vez que es ejecutado.

Al comparar los mejores modelos obtenidos: GeoTransform21 para small-HiTS y GeoTransform17 para small-ALeRCE-TFv1, se obtienen los modelos con más alto AUROC promedio del estudio, con un valor de 99.42 para small-HiTS y 94.14 para small-ALeRCE-ZTFv1. Las mejoras sobre el algoritmo original de GeoTransform son significativas tanto estadísticamente como en la práctica. Su potencial para identificar muestras tipo bogus de forma automática resulta llamativa para el etiquetado de bogus sin la necesidad de expertos humanos. Con esta comparación se cumple el objetivo O4 de la tesis, de comparar las mejoras propuestas a GeoTransform con otros modelos del estado del arte. Pese a que los resultados reportados son los más altos sobre el conjunto de test, se ha de tener en cuenta que estos no

pueden llegar y ser declarados como los mejores modelos, ya que se puede estar en presencia de un efecto de sobre-ajuste al conjunto de test, particularmente después de la cantidad de experimentos realizados. A pesar de que el mismo flujo de selección de transformaciones sobre GeoTransform99 obtuvo el mejor resultado en ambas bases de datos, lo que es un buen indicador de la robustez del método, se han de seleccionar modelos con precaución, debido a la falta de una estrategia de validación para modelos de detección de anomalías que no incluyan a las anomalías en si, a las cuales se supone no se tiene acceso más que en conjuntos de test. Pruebas de estos modelos en más bases de datos podrían ayudar a tomar una decisión más informada de si son los mejores, sin embargo se necesita otra metodología con sustento teórico para tomar esta decisión adecuadamente.

## 6.3. Detección automática de bogus para la clasificación de objetos astronómicos

Durante el desarrollo del stamp classifier una de las principales dificultades fue la adquisición de bogus etiquetados por expertos, por lo que en vista del buen funcionamiento para etiquetar bogus de GeoTransform, el que solo es entrenado con *inliers*, se prueba su efectividad como método automático para el etiquetado de bogus con los que luego entrenar el stamp classifier sin metadatos. Para evidenciar la efectividad de los bogus entrenados con GeoTransform estos son comparados con utilizar bogus de una etapa temprana de ZTF y bogus de una etapa avanzada de ZTF. Para obtener bogus a partir de GeoTransform se usan 3 modelos diferentes, debido a la incapacidad de escoger un modelo como el mejor sin utilizar el conjunto de test. El desempeño obtenido del stamp classifier sin metadatos al ser entrenado con los diferentes tipos de bogus muestra que utilizar los bogus etiquetados con cualquier modelo de GeoTransform supera el utilizar bogus de etapa temprana, logrando una mejora de hasta 7.74 puntos de *accuracy* en promedio. Con este resultado se puede declarar de forma parcial el cumplimiento de la hipótesis, donde efectivamente se logran etiquetar bogus con GeoTransform de forma automática, sin embargo, la calidad de estos bogus no es la suficiente para equiparar el desempeño obtenido al utilizar bogus etiquetados por expertos en etapas avanzadas de un estudio astronómico como ZTF, pero si resulta útil para superar con creces el desempeño de contar con pocos bogus etiquetados en etapas tempranas de un estudio astronómico. Con esto también se logra de forma parcial el O5 de esta tesis, de equiparar el desempeño de usar bogus etiquetados por expertos con bogus de GeoTransform al entrenar el stamp classifier sin metadatos. Se demuestra de forma satisfactoria que la utilización de GeoTransform como herramienta de etiquetado automático de bogus es de utilidad especialmente para etapas tempranas de estudios astronómicos, lo que resultará crucial en futuros estudios astronómicos.

## 6.4. Resumen de contribuciones

- Desarrollar el stamp classifier, modelo que permite clasificar entre 5 clases de objetos astronómicos (VS, AGN, SN, asteroides y bogus) utilizando sólo la primera alerta detectada para un objeto.
- Se incluye un regularizador de entropía en la función de costo del stamp classifier que permite obtener una mejor interpretabilidad de las probabilidades predichas por el

modelo para cada clase.

- La implementación del stamp classifier permitió la creación del SN Hunter, herramienta con que astrónomos de ALeRCE han logrado identificar 6846 candidatos a SN nunca antes vistos.
- Nuevos esquemas de entrenamiento para GeoTransform, uno basado en detención temprana para evitar sobre-ajuste y la proposición preliminar de otro que permite su entrenamiento sobre grandes bases de datos sin utilizar grandes cantidades de memoria RAM.
- Inclusión de transformaciones de kernel Gaussiano y Laplaciano a GeoTransform, que mejoran las detección de bogus.
- Creación de una metodología empírica para la selección de transformaciones que permitió crear 5 métodos distintos de selección de transformaciones, donde cada uno mejora el desempeño de GeoTransform.
- Mejorar el estado del arte del desempeño de GeoTransform en bases de datos astronómicas.
- Demostrar que el etiquetado de bogus de forma automática con GeoTransform permite mejorar el desempeño de clasificadores de imágenes astronómicas en etapas tempranas de un estudio astronómico.

## 6.5. Últimos comentarios

Como se menciona a lo largo del trabajo, actualmente el stamp classifier es una herramienta activamente utilizada por el broker ALeRCE, el cual está en continuo proceso de mejora para su utilización en nuevos estudios astronómicos como ATLAS y LSST. El tesista está participando de forma activa en este proceso de mejora continua como parte del equipo de investigación de ALeRCE.

Dado los experimentos exitosos de utilizar GeoTransform como herramienta de etiquetado automático de bogus en etapas tempranas de un estudio astronómico, es que se contempla incorporar esta estrategia en producción como una herramienta más de ALeRCE, para aliviar la carga de etiquetado por expertos en futuros estudios astronómicos.

En el estudio de detección de anomalías, se abren muchas avenidas de trabajo futuro en lo que respecta a los conjuntos adecuados de transformaciones para GeoTransform. Los resultados mostrados en esta tesis fueron aplicados sobre bases de datos astronómicos, por lo cual resta evidenciar si es que los resultados e hipótesis comprobadas se mantienen para datos de otros dominios.

En general, en esta tesis se abordaron 3 problemas diferentes, pero con muchos elementos en común, logrando no sólo abrir posibilidades de investigación futura, sino que también se han desarrollado herramientas prácticas, a partir de las cuales la astronomía puede beneficiarse para mejorar los procesos en que realiza ciencia sobre grandes cantidades de datos.

# Bibliografía


[1] P. Sánchez-Sáez, I. Reyes, C. Valenzuela, F. Förster, S. Eyheramendy, F. Elorrieta, F. Bauer, G. Cabrera-Vives, P. Estévez, M. Catelan *et al.*, "Alert classification for the alerce broker system: The light curve classifier," *The Astronomical Journal*, vol. 161, no. 3, p. 141, 2021.

[2] R. Carrasco-Davis, E. Reyes, C. Valenzuela, F. Förster, P. Estévez, G. Pignata, F. E. Bauer, I. Reyes, P. Sánchez-Sáez, G. Cabrera-Vives *et al.*, "Alert classification for the alerce broker system: The real-time stamp classifier," *arXiv preprint arXiv:2008.03309*, 2021.

[3] F. Förster, G. Cabrera-Vives, E. Castillo-Navarrete, P. Estévez, P. Sánchez-Sáez, J. Arredondo, F. Bauer, R. Carrasco-Davis, M. Catelan, F. Elorrieta *et al.*, "The automatic learning for the rapid classification of events (alerce) alert broker," *The Astronomical Journal*, vol. 161, no. 5, p. 242, 2021.

[4] E. Reyes and P. Estévez, "Transformation based deep anomaly detection in astronomical images," in *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2020, pp. 1–8.

[5] J. Arredondo, "Clasificación de objetos astronómicos mediante sus primeras alertas utilizando aprendizaje profundo," *Tesis de magíster*, Universidad de Santiago de Chile, 2021.

[6] S. Ivezić, Žand Kahn, J. Tyson, B. Abel, E. Acosta, R. Allsman, D. Alonso, Y. AlSayyad, S. Anderson, J. Andrew *et al.*, "Lsst: from science drivers to reference design and anticipated data products," *The Astrophysical Journal*, vol. 873, no. 2, p. 111, 2019.

[7] R. Smith, R. Dekany, C. Bebek, E. Bellm, K. Bui, J. Cromer, P. Gardner, M. Hoff, S. Kaye, S. Kulkarni *et al.*, "The zwicky transient facility observing system," in *Ground-based and Airborne Instrumentation for Astronomy V*, vol. 9147. International Society for Optics and Photonics, 2014, p. 914779.

[8] A. Riess, A. Filippenko, P. Challis, A. Clocchiatti, A. Diercks, P. Garnavich, R. Gilliland, C. Hogan, S. Jha, R. Kirshner *et al.*, "Observational evidence from supernovae for an accelerating universe and a cosmological constant," *The Astronomical Journal*, vol. 116, no. 3, p. 1009, 1998.

[9] B. Schmidt, N. Suntzeff, M. Phillips, R. Schommer, A. Clocchiatti, R. Kirshner, P. Garnavich, P. Challis, B. Leibundgut, J. Spyromilio *et al.*, "The high-z supernova search: measuring cosmic deceleration and global curvature of the universe using type ia supernovae," *The Astrophysical Journal*, vol. 507, no. 1, p. 46, 1998.

[10] T. Matheson, A. Saha, R. Snodgrass, and J. Kececioglu, "Antares: The arizona-noao temporal analysis and response to events system," *Hot-Wiring the Transient Universe*, vol. 3, 2014.

[11] K. Smith, "Lasair: the transient alert broker for lsst: Uk," *The Extragalactic Explosive Universe: the New Era of Transient Surveys and Data-Driven Discovery*, p. 51, 2019.

[12] I. Barandiaran, "The random subspace method for constructing decision forests," *IEEE Trans. Pattern Anal. Mach. Intell*, vol. 20, no. 8, pp. 1–22, 1998.

[13] T. Naylor, "An optimal extraction algorithm for imaging photometry," *Monthly Notices of the Royal Astronomical Society*, vol. 296, no. 2, pp. 339–346, 1998.

[14] Q. Zhao, J. Sun, C. Yu, C. Cui, L. Lv, and J. Xiao, "A paralleled large-scale astronomical cross-matching function," in *Int. Conf. Algorithms Architetures Parallel Processing (ICA3PP)*. Springer, 2009, pp. 604–614.

[15] F. Liu, K. Ting, and Z.-H. Zhou, "Isolation forest," in *2008 Eighth IEEE Int. Conf. Data Mining (ICDM)*. IEEE, 2008, pp. 413–422.

[16] L. Manevitz and M. Yousef, "One-class svms for document classification," *Journal of machine Learning research*, vol. 2, no. Dec, pp. 139–154, 2001.

[17] F. Korn, B.-U. Pagel, and C. Faloutsos, "On the "dimensionality curse" and the "self-similarity blessing"," *IEEE Transactions on Knowledge and Data Engineering*, vol. 13, no. 1, pp. 96–111, 2001.

[18] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, "Deep learning for computer vision: A brief review," *Computational intelligence and neuroscience*, vol. 2018, 2018.

[19] M. Sakurada and T. Yairi, "Anomaly detection using autoencoders with nonlinear dimensionality reduction," in *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis*. ACM, 2014, p. 4.

[20] S. Akcay, A. Atapour-Abarghouei, and T. Breckon, "Ganomaly: Semi-supervised anomaly detection via adversarial training," in *Asian Conference on Computer Vision*. Springer, 2018, pp. 622–637.

[21] I. Golan and R. El-Yaniv, "Deep anomaly detection using geometric transformations," in *Advances in Neural Information Processing Systems*, 2018, pp. 9758–9769.

[22] C. Huang, J. Cao, F. Ye, M. Li, Y. Zhang, and C. Lu, "Inverse-transform autoencoder for anomaly detection," *arXiv preprint arXiv:1911.10676*, 2019.

[23] E. Reyes, P. Estévez, I. Reyes, G. Cabrera-Vives, P. Huijse, R. Carrasco, and F. Forster, "Enhanced rotational invariant convolutional neural network for supernovae detection," in *2018 Int. Jt. Conf. Neural Netw. (IJCNN)*. IEEE, 2018, pp. 1–8.

[24] R. Carrasco-Davis, G. Cabrera-Vives, F. Förster, P. Estévez, P. Huijse, P. Protopapas, I. Reyes, J. Martínez-Palomera, and C. Donoso, "Deep learning for image sequence classification of astronomical events," *Publications of the Astronomical Society of the Pacific*, vol. 131, no. 1004, p. 108006, 2019.

[25] G. Cabrera-Vives, I. Reyes, F. Förster, P. Estévez, and J.-C. Maureira, "Deep-hits: Rotation invariant convolutional neural network for transient detection," *The Astrophysical Journal*, vol. 836, no. 1, p. 97, 2017.

[26] F. Förster, J. Maureira, J. San Martín, M. Hamuy, J. Martínez, P. Huijse, G. Cabrera, L. Galbany, T. De Jaeger, S. González-Gaitán *et al.*, "The high cadence transient survey (hits). i. survey design and supernova shock breakout constraints," *The Astrophysical Journal*, vol. 832, no. 2, p. 155, 2016.

[27] S. Kent, "Sloan digital sky survey," *Astrophysics and Space Science*, vol. 217, no. 1, pp. 27–30, 1994.

[28] C. Alcock, R. Allsman, D. Alves, T. Axelrod, A. Becker, D. Bennett, K. Cook, A. Drake, K. Freeman, M. Geha *et al.*, "The macho project: microlensing detection efficiency," *The Astrophysical Journal Supplement Series*, vol. 136, no. 2, p. 439, 2001.

[29] J. Tonry, L. Denneau, A. Heinze, B. Stalder, K. Smith, S. Smartt, C. Stubbs, H. Weiland, and A. Rest, "Atlas: a high-cadence all-sky survey system," *Publications of the Astronomical Society of the Pacific*, vol. 130, no. 988, p. 064505, 2018.

[30] R. Carrasco-Davis, "Image sequence simulation and deep learning for astronomical object classification," *Master's thesis*, Universidad de Chile, 2019.

[31] S. Hardesty, "Seds messier database (messier catalog)," *Reference Reviews*, vol. 19, no. 8, pp. 45–45, 2005.

[32] B. Lasker, M. Lattanzi, B. McLean, B. Bucciarelli, R. Drimmel, J. Garcia, G. Greene, F. Guglielmetti, C. Hanley, G. Hawkins *et al.*, "The second-generation guide star catalog: description and properties," *The Astronomical Journal*, vol. 136, no. 2, p. 735, 2008.

[33] N. Zacharias, S. Urban, M. Zacharias, G. Wycoff, D. Hall, D. Monet, and T. Rafferty, "The second us naval observatory ccd astrograph catalog (ucac2)," *The Astronomical Journal*, vol. 127, no. 5, p. 3043, 2004.

[34] M. Nieto-Santisteban, A. Thakar, and A. S. Szalay, "Cross-matching very large datasets," in *National Science and Technology Council (NSTC) NASA Conference*. Citeseer, 2007.

[35] H. Kerner, D. Wellington, K. Wagstaff, J. Bell, C. Kwan, and H. Amor, "Novelty detec-

tion for multispectral images with application to planetary exploration," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 9484–9491.

[36] M. Bannister, A. Bhandare, P. Dybczyński, A. Fitzsimmons, A. Guilbert-Lepoutre, R. Jedicke, M. Knight, K. Meech, A. McNeill, S. Pfalzner *et al.*, "The natural history of "oumuamua"," *Nature Astronomy*, vol. 3, no. 7, pp. 594–602, 2019.

[37] J. Forbes and A. Loeb, "Turning up the heat on "oumuamua"," *The Astrophysical Journal Letters*, vol. 875, no. 2, p. L23, 2019.

[38] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press Cambridge, 2016, vol. 1.

[39] F. Rosenblatt, "The perceptron: a probabilistic model for information storage and organization in the brain." *Psychological review*, vol. 65, no. 6, p. 386, 1958.

[40] V. Nair and G. Hinton, "Rectified linear units improve restricted boltzmann machines," in *ICML*, 2010.

[41] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*. PMLR, 2015, pp. 448–456.

[42] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[43] D. Hendrycks and K. Gimpel, "A baseline for detecting misclassified and out-of-distribution examples in neural networks," *arXiv preprint arXiv:1610.02136*, 2016.

[44] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM computing surveys (CSUR)*, vol. 41, no. 3, pp. 1–58, 2009.

[45] V. Hodge and J. Austin, "A survey of outlier detection methodologies," *Artificial intelligence review*, vol. 22, no. 2, pp. 85–126, 2004.

[46] M. Pimentel, D. Clifton, L. Clifton, and L. Tarassenko, "A review of novelty detection," *Signal Processing*, vol. 99, pp. 215–249, 2014.

[47] T. Ince, S. Kiranyaz, L. Eren, M. Askar, and M. Gabbouj, "Real-time motor fault detection by 1-d convolutional neural networks," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 11, pp. 7067–7075, 2016.

[48] R. Chalapathy and S. Chawla, "Deep learning for anomaly detection: A survey," *arXiv preprint arXiv:1901.03407*, 2019.

[49] C. Aggarwal, "Outlier analysis," in *Data mining*. Springer, 2015, pp. 237–263.

[50] H. Wang, M. Bah, and M. Hammad, "Progress in outlier detection techniques: A survey," *IEEE Access*, vol. 7, pp. 107 964–108 000, 2019.

[51] L. Ruff, J. Kauffmann, R. A. Vandermeulen, G. Montavon, W. Samek, M. Kloft, T. Dietterich, and K.-R. Müller, "A unifying review of deep and shallow anomaly detection," *Proceedings of the IEEE*, 2021.

[52] N. Görnitz, M. Kloft, K. Rieck, and U. Brefeld, "Toward supervised anomaly detection," *Journal of Artificial Intelligence Research*, vol. 46, pp. 235–262, 2013.

[53] L. Ruff, R. Vandermeulen, N. Gornitz, A. Binder, E. Muller, and M. Kloft, "Deep support vector data description for unsupervised and semi-supervised anomaly detection," in *Proceedings of the ICML 2019 Workshop on Uncertainty and Robustness in Deep Learning, Long Beach, CA, USA*, 2019, pp. 9–15.

[54] M. Goldstein and S. Uchida, "A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data," *PloS one*, vol. 11, no. 4, p. e0152173, 2016.

[55] B. Schölkopf, J. Platt, J. Shawe-Taylor, A. Smola, and R. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Computation*, vol. 13, no. 7, pp. 1443–1471, 2001.

[56] G. Hinton, A. Krizhevsky, and S. Wang, "Transforming auto-encoders," in *Int. Conf. Artif. Neural Netw. (ICANN)*. Springer, 2011, pp. 44–51.

[57] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems*, 2014, pp. 2672–2680.

[58] D. Tax, "One-class classification: Concept learning in the absence of counter-examples," *Technische Universiteit Delft*, Doctoral Thesis, 2002.

[59] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," in *Int. Conf. Learning Representations (ICLR)*, 2016.

[60] S. Zhai, Y. Cheng, W. Lu, and Z. Zhang, "Deep structured energy based models for anomaly detection," in *Int. Conf. Machine Learning (ICML)*, 2016, pp. 1100–1109.

[61] L. Deecke, R. Vandermeulen, L. Ruff, S. Mandt, and M. Kloft, "Image anomaly detection with generative adversarial networks," in *Computers European Conference (ECML PKDD)*. Springer, 2018, pp. 3–17.

[62] Y. Liu, Z. Li, C. Zhou, Y. Jiang, J. Sun, M. Wang, and X. He, "Generative adversarial active learning for unsupervised outlier detection," *IEEE T. Knowl. Data En.*, 2019.

[63] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," *University of Toronto*, (Technical Report), 2009.

[64] N. Wicker, J. Muller, R. Kalathur, and O. Poch, "A maximum likelihood approximation method for dirichlet's parameter estimation," *Comput. Stat. Data An.*, vol. 52, no. 3, pp. 1315–1322, 2008.

[65] T. Minka, "Estimating a dirichlet distribution," *Technical report, MIT*, 2000.

[66] S. Zagoruyko and N. Komodakis, "Wide residual networks," in *Proc. British Machine Vision Conf. (BMVC)*, 2016, pp. 87.1–87.12.

[67] L. Jing and Y. Tian, "Self-supervised visual feature learning with deep neural networks: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[68] M. Noroozi and P. Favaro, "Unsupervised learning of visual representations by solving jigsaw puzzles," in *European conference on computer vision*. Springer, 2016, pp. 69–84.

[69] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. Efros, "Context encoders: Feature learning by inpainting," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2536–2544.

[70] N. Komodakis and S. Gidaris, "Unsupervised representation learning by predicting image rotations," in *International Conference on Learning Representations (ICLR)*, 2018.

[71] Y. LeCun and I. Misra, "Self-supervised learning: The dark matter of intelligence," *Facebook AI*, (https://ai.facebook.com/blog/self-supervised-learning-the-dark-matter-of-intelligence/) Mar 2021. [Online]. Available: https://ai.facebook.com/blog/self-supervised-learning-the-dark-matter-of-intelligence/

[72] Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato, and F. Huang, "A tutorial on energy-based learning," *Predicting structured data*, vol. 1, no. 0, 2006.

[73] D. Hendrycks, M. Mazeika, and T. Dietterich, "Deep anomaly detection with outlier exposure," in *International Conference on Learning Representations*, 2018.

[74] D. Hendrycks, M. Mazeika, S. Kadavath, and D. Song, "Using self-supervised learning can improve model robustness and uncertainty," *Advances in Neural Information Processing Systems*, vol. 32, pp. 15 663–15 674, 2019.

[75] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[76] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[77] I. Nun, P. Protopapas, B. Sim, M. Zhu, R. Dave, N. Castro, and K. Pichara, "Fats: Feature analysis for time series," *arXiv preprint arXiv:1506.00010*, 2015.

[78] B. Naul, J. Bloom, F. Pérez, and S. van der Walt, "A recurrent neural network for classification of unevenly sampled variable stars," *Nature Astronomy*, vol. 2, no. 2, pp. 151–155, 2018.

[79] A. Mahabal, K. Sheth, F. Gieseke, A. Pai, S. Djorgovski, A. Drake, and M. Graham, "Deep-learnt classification of light curves," in *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 2017, pp. 1–8.

[80] S. Dieleman, K. Willett, and J. Dambre, "Rotation-invariant convolutional neural networks for galaxy morphology prediction," *Monthly notices of the royal astronomical society*, vol. 450, no. 2, pp. 1441–1459, 2015.

[81] M. Pérez-Carrasco, G. Cabrera-Vives, M. Martinez-Marin, P. Cerulo, R. Demarco, P. Protopapas, J. Godoy *et al.*, "Multiband galaxy morphologies for clash: a convolutional neural network transferred from candels," *Publications of the Astronomical Society of the Pacific*, vol. 131, no. 1004, p. 108002, 2019.

[82] P. Barchi, R. de Carvalho, R. Rosa, R. Sautter, M. Soares-Santos, B. Marques, E. Clua, T. Gonçalves, C. de Sá-Freitas, and T. Moura, "Machine and deep learning applied to galaxy morphology-a comparative study," *Astronomy and Computing*, vol. 30, p. 100334, 2020.

[83] C. Gómez, M. Neira, M. Hernández Hoyos, P. Arbeláez, and J. Forero-Romero, "Classifying image sequences of astronomical transients with deep neural networks," *Monthly Notices of the Royal Astronomical Society*, vol. 499, no. 3, pp. 3130–3138, 2020.

[84] D. Goldstein, C. D'Andrea, J. Fischer, R. Foley, R. Gupta, R. Kessler, A. Kim, R. Nichol, P. Nugent, A. Papadopoulos *et al.*, "Automated transient identification in the dark energy survey," *The Astronomical Journal*, vol. 150, no. 3, p. 82, 2015.

[85] G. Cabrera-Vives, I. Reyes, F. Förster, P. Estévez, and J.-C. Maureira, "Supernovae detection by using convolutional neural networks," in *2016 Int. Jt. Conf. Neural Netw. (IJCNN)*. IEEE, 2016, pp. 251–258.

[86] D. Duev, A. Mahabal, F. Masci, M. Graham, B. Rusholme, R. Walters, I. Karmarkar, S. Frederick, M. Kasliwal, and U. o. Rebbapragada, "Real-bogus classification for the zwicky transient facility using deep learning," *Monthly Notices of the Royal Astronomical Society*, vol. 489, no. 3, pp. 3582–3590, 2019.

[87] D. Turpin, M. Ganet, S. Antier, E. Bertin, L. Xin, N. Leroy, C. Wu, Y. Xu, X. Han, H. Cai *et al.*, "Vetting the optical transient candidates detected by the gwac network using convolutional neural networks," *Monthly Notices of the Royal Astronomical Society*, vol. 497, no. 3, pp. 2641–2650, 2020.

[88] K. Malanchev, A. Volnova, M. Kornilov, M. Pruzhinskaya, E. Ishida, F. Mondon, and V. Korolev, "Use of machine learning for anomaly detection problem in large astronomical databases." in *DAMDID/RCDL*, 2019, pp. 205–216.

[89] M. Lochner and B. Bassett, "Astronomaly: Personalised active anomaly detection in astronomical data," *Astronomy and Computing*, p. 100481, 2021.

[90] M. D'Addona, G. Riccio, S. Cavuoti, C. Tortora, and M. Brescia, "Anomaly detection in astrophysics: a comparison between unsupervised deep and machine learning on kids data," *arXiv preprint arXiv:2006.08235*, 2020.

[91] H.-W. Lin, Y.-T. Chen, J.-H. Wang, S.-Y. Wang, F. Yoshida, W.-H. Ip, S. Miyazaki, and T. Terai, "Machine-learning-based real–bogus system for the hsc-ssp moving object

detection pipeline," *Publ. Astron. Soc. Jpn.*, vol. 70, no. SP1, p. S39, 2018.

[92] K. Storey-Fisher, M. Huertas-Company, N. Ramachandra, F. Lanusse, A. Leauthaud, Y. Luo, and S. Huang, "Anomaly detection in astronomical images with generative adversarial networks," *arXiv preprint arXiv:2012.08082*, 2020.

[93] E. Bellm, S. Kulkarni, M. Graham, R. Dekany, R. Smith, R. Riddle, F. Masci, G. Helou, T. Prince, S. Adams *et al.*, "The zwicky transient facility: System overview, performance, and first results," *Publications of the Astronomical Society of the Pacific*, vol. 131, no. 995, p. 018002, 2018.

[94] F. Masci, R. Laher, B. Rusholme, D. Shupe, S. Groom, J. Surace, E. Jackson, S. Monkewitz, R. Beck, D. Flynn *et al.*, "The zwicky transient facility: Data processing, products, and archive," *Publ. Astron. Soc. Pac.*, vol. 131, no. 995, p. 018003, 2018.

[95] T. Jayasinghe, C. Kochanek, K. Stanek, B. Shappee, T. W. Holoien, T. A. Thompson, J. Prieto, S. Dong, M. Pawlak, J. Shields *et al.*, "The asas-sn catalogue of variable stars i: The serendipitous survey," *Monthly Notices of the Royal Astronomical Society*, vol. 477, no. 3, pp. 3145–3163, 2018.

[96] T. Jayasinghe, K. Stanek, C. Kochanek, B. Shappee, T. W. Holoien, T. A. Thompson, J. Prieto, S. Dong, M. Pawlak, O. Pejcha *et al.*, "The asas-sn catalogue of variable stars–ii. uniform classification of 412 000 known variables," *Monthly Notices of the Royal Astronomical Society*, vol. 486, no. 2, pp. 1907–1943, 2019.

[97] ——, "The asas-sn catalogue of variable stars iii: variables in the southern tess continuous viewing zone," *Monthly Notices of the Royal Astronomical Society*, vol. 485, no. 1, pp. 961–971, 2019.

[98] ——, "The asas-sn catalogue of variable stars–v. variables in the southern hemisphere," *Monthly Notices of the Royal Astronomical Society*, vol. 491, no. 1, pp. 13–28, 2020.

[99] E. Massaro, A. Maselli, C. Leto, P. Marchegiani, M. Perri, P. Giommi, and S. Piranomonte, "The 5th edition of the roma-bzcat. a short presentation," *Astrophysics and Space Science*, vol. 357, no. 1, pp. 1–4, 2015.

[100] E. Flesch, "Vizier online data catalog: The half million quasars (hmq) catalogue (flesch, 2015)," *VizieR Online Data Catalog*, pp. VII–273, 2015.

[101] ——, "The million quasars (milliquas) catalogue, v6. 4," *arXiv preprint arXiv:1912.05614*, 2019.

[102] K. Oh, K. Sukyoung, K. Schawinski, M. Koss, B. Trakhtenbrot, and K. Soto, "A new catalog of type 1 agns and its implications on the agn unified model," *The Astrophysical Journal Supplement Series*, vol. 219, no. 1, p. 1, 2015.

[103] A. Drake, M. Graham, S. Djorgovski, M. Catelan, A. Mahabal, G. Torrealba, D. García-Álvarez, C. Donalek, J. Prieto, R. Williams *et al.*, "The catalina surveys periodic variable star catalog," *The Astrophysical Journal Supplement Series*, vol. 213, no. 1, p. 9,

2014.

[104] A. Drake, S. Djorgovski, M. Catelan, M. Graham, A. Mahabal, S. Larson, E. Christensen, G. Torrealba, E. Beshore, R. McNaught *et al.*, "The catalina surveys southern periodic variable star catalogue," *Monthly Notices of the Royal Astronomical Society*, vol. 469, no. 3, pp. 3688–3712, 2017.

[105] L. Palaversa, Ž. Ivezić, L. Eyer, D. Ruždjak, D. Sudar, M. Galin, A. Kroflin, M. Mesarić, P. Munk, D. Vrbanec *et al.*, "Exploring the variable sky with linear. iii. classification of periodic light curves," *The Astronomical Journal*, vol. 146, no. 4, p. 101, 2013.

[106] N. Mowlavi, I. Lecoeur-Taïbi, T. Lebzelter, L. Rimoldini, D. Lorenz, M. Audard, J. De Ridder, L. Eyer, L. Guy, B. Holl *et al.*, "Gaia data release 2-the first gaia catalogue of long-period variable candidates," *Astronomy & Astrophysics*, vol. 618, p. A58, 2018.

[107] L. Rimoldini, B. Holl, M. Audard, N. Mowlavi, K. Nienartowicz, D. Evans, L. Guy, I. Lecoeur-Taïbi, G. J. de Fombelle, O. Marchal *et al.*, "Gaia data release 2-all-sky classification of high-amplitude pulsating stars," *Astronomy & Astrophysics*, vol. 625, p. A97, 2019.

[108] M. Wenger, F. Ochsenbein, D. Egret, P. Dubois, F. Bonnarel, S. Borde, F. Genova, G. Jasniewicz, S. Laloë, S. Lesteven *et al.*, "The simbad astronomical database-the cds reference database for astronomical objects," *Astronomy and Astrophysics Supplement Series*, vol. 143, no. 1, pp. 9–22, 2000.

[109] P. Warde, "How many images do you need to train a neural network?" (https://petewarden.com/2017/12/14/how-many-images-do-you-need-to-train-a-neural-network/) Oct 2021. [Online]. Available: https://petewarden.com/2017/12/14/how-many-images-do-you-need-to-train-a-neural-network/

[110] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[111] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek, "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation," *PloS one*, vol. 10, no. 7, p. e0130140, 2015.

[112] S. Dieleman, J. De Fauw, and K. Kavukcuoglu, "Exploiting cyclic symmetry in convolutional neural networks," in *International conference on machine learning*. PMLR, 2016, pp. 1889–1898.

[113] G. Pereyra, G. Tucker, J. Chorowski, L. Kaiser, and G. Hinton, "Regularizing neural networks by penalizing confident output distributions," *arXiv preprint arXiv:1701.06548*, 2017.

[114] L. McInnes, J. Healy, N. Saul, and L. Großberger, "Umap: Uniform manifold approximation and projection," *Journal of Open Source Software*, vol. 3, no. 29, 2018.

[115] D. Pal, S. Nallamothu, and M. Savvides, "Towards a hypothesis on visual transformation based self-supervision," *arXiv preprint arXiv:1911.10594*, 2019.

[116] R. Gonzalez and R. Woods, "Digital image processing, 3rd ed prentice hall," *Upper Saddle River, New Jersey*, 2008.

[117] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.

[118] C. Shannon, "A mathematical theory of communication," *The Bell system technical journal*, vol. 27, no. 3, pp. 379–423, 1948.

[119] J. Vergara and P. Estévez, "A review of feature selection methods based on mutual information," *Neural computing and applications*, vol. 24, no. 1, pp. 175–186, 2014.

[120] N. Tapia and P. Estévez, "On the information plane of autoencoders," in *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2020, pp. 1–8.

[121] K. Wickstrøm, S. Løkse, M. Kampffmeyer, S. Yu, J. Principe, and R. Jenssen, "Information plane analysis of deep neural networks via matrix-based renyi's entropy and tensor kernels," *arXiv preprint arXiv:1909.11396*, 2019.

[122] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," *Advances in neural information processing systems*, vol. 30, 2017.

[123] J. Bergstra, D. Yamins, and D. Cox, "Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures," in *International conference on machine learning*. PMLR, 2013, pp. 115–123.

[124] E. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. Le, "Autoaugment: Learning augmentation policies from data," *arXiv preprint arXiv:1805.09501*, 2018.

[125] R. Sutton and A. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[126] E. Bertin and S. Arnouts, "Sextractor: Software for source extraction," *Astronomy and astrophysics supplement series*, vol. 117, no. 2, pp. 393–404, 1996.

[127] P. Stetson, "Daophot: A computer program for crowded-field stellar photometry," *Publications of the Astronomical Society of the Pacific*, vol. 99, no. 613, p. 191, 1987.

# Apéndice A

# Clasificación temprana de objetos astronómicos

Tabla A.1: Lista de los 23 metadatos de las alertas de ZTF usadas como entrada al stamp classifier. Las definiciones de esta tabla provienen de la documentación de los datos de ZTF[a].

| **Metadato** | **Descripción** [unidades] |
|---|---|
| `sgscore{1, 2, 3}` | Puntaje de estrella/galaxia de la {primera, segunda, tercera} fuente más cercana del catálogo PanSTARRS1, $0 \leq$ `sgscore` $\leq 1$, donde un valor de 1 significa mayor probabilidad de ser una estrella, -999 cuando no hay fuente. |
| `distpsnr {1, 2, 3}` | Distancia de la {primera, segunda, tercera} fuente más cercana del catálogo PanSTARRS1, si existe una dentro de 30 arcosegundos, -999 si es que no hay fuente [arcosegundos]. |
| `isdiffpos` | t (convertido a 1) si el candidato es de una substracción positiva (ciencia menos referencia); f (convertido a 0) si el candidato es una substracción negativa (referencia menos ciencia). |
| `fwhm` | Full Width Half Max (propiedad de distribución Gaussiana) asumiendo un centro gaussiano de la alerta del candidato en la imagen de ciencia de SExtractor [126] [pixels]. |
| `magpsf` | Magnitud del ajuste de PSF fotométrico de la alerta en la imagen de diferencia [mag]. |
| `sigmapsf` | Incertidumbre de 1-sigma en `magpsf` [mag]. |
| `ra`, `dec` | Coordenadas *Right ascension* y declinación del candidato; J2000 [deg]. |
| `diffmaglim` | Mag. lím. de 5-sigma en imagen de diferencia, basado en ajuste fotométrico de PSF [mag]. |
| `classtar` | Puntaje de clasificación estrella/galaxia de la alerta en imagen de diferencia, de SExtractor. |
| `ndethist` | Número de detecciones espacialmente coincidentes que caen dentro de 1.5 arcosegundos al ir hacia el comienzo del estudio astronómico; sólo son contadas las detecciones que caen sobre el mismo campo e identificador (ID) de canal de lectura donde el candidato de entrada es observado. Todas las detecciones crudas hasta un nivel fotométrico de señal/ruido$\approx$3 son incluidas. |
| `ncovhist` | Número de veces que la posición del candidato de entrada cae en cualquiera de las regiones y canal de lectura, en retrospectiva al comienzo del estudio astronómico. |
| `chinr`, `sharpnr` | DAOPhot [127] parámetros chi, sharp de la fuente más cercana en la imagen de referencia catálogo-PSF dentro de 30 arcosegundos. |
| Coordenadas eclípticas | latitud y longitud eclíptica calculadas desde los metadatos `ra`, `dec` del candidato [grados]. |
| Coordenadas galácticas | latitud y longitud galáctica calculadas desde los metadatos `ra`, `dec` del candidato [grados]. |
| no-detecciones aprox. | `ncovhist` menos `ndethist`. Número aproximado de observaciones en la posición del candidato, con una señal más baja que la razón señal/ruido$\approx$3. |

[a] `https://zwickytransientfacility.github.io/ztf-avro-alert/`

Tabla A.2: Valores límite para cada metadato. "max" o "min" en el rango de valores límite por cada metadato significa que el valor máximo y mínimo se mantiene para el respectivo metadato, respectivamente.

| Metadato | [valor min, valor max] |
|---|---|
| sgscore1 | [-1, max] |
| distpsnr1 | [-1, max] |
| sgscore2 | [-1, max] |
| distpsnr2 | [-1, max] |
| sgscore3 | [-1, max] |
| distpsnr3 | [-1, max] |
| ifwhm | [min, 10] |
| ndethist | [min, 20] |
| ncovhist | [min, 3000] |
| chinr | [-1, 15] |
| sharpnr | [-1, 1.5] |
| no-detecciones | [min, 2000] |

## A.1. Comparación con otros modelos de la literatura

Como se menciona en la Sección 5.1 del capítulo de Discusión, el stamp classifier es el primer modelo en la literatura donde se clasifican 5 clases de objetos astronómicos, por lo cual es difícil compararlo con otros modelos, al actualmente no existir un trabajo publicado que realice la misma tarea. En un intento por evaluar la validez del stamp classifier, en [2] se mide el grado de congruencia entre las clasificaciones realizadas por el stamp classifier y el LCC de ALeRCE [1] sobre un conjunto de 186,794 alertas de ZTF no etiquetadas, estas alertas provienen de objetos con más de una detección (SNe, AGN y VS). El LCC utiliza como modelo de clasificación un RF entrenado sobre características extraídas a partir de las curvas de luz de los objetos astronómicos, y requiere al menos 6 detecciones en una de las dos bandas (rangos del espectro de luz) de observación de ZTF. Debido a la utilización de curvas de luz, las que permiten capturar la dinámica de un objeto astronómico en el tiempo, se presume que el LCC es capaz de identificar objetos de forma más confiable que el stamp classifier, al ser capaz de caracterizarlos en clases más específicas que las 5 clases del stamp classifier. Por ejemplo, es capaz de clasificar SNe en sus diferentes tipos, como lo son SNIa, SNII, etc. Por esto es que se busca validar el stamp classifier al medir qué tanto coinciden sus clasificaciones con las del LCC sobre el conjunto de alertas no etiquetadas. La comparación de ambos modelos puede verse en la Tabla A.3, donde se muestran las clases asignadas por el stamp classifier (filas) y el LCC (columnas) a las alertas del conjunto no etiquetado. De esta tabla se encuentra que el stamp classifier clasifica como SN el 78 % de las alertas que el LCC dice que son SNe, 85 % de los AGNs y 96 % de las VS. Si es que se consideran las clasificaciones del LCC como etiquetas reales, entonces los porcentajes anteriores serían *recalls* por clase del stamp classifier.

Con el LCC no es posible identificar objetos con menos de 6 detecciones, por lo cual no se puede evaluar el desempeño en clases como bogus o asteroides, los que rara vez generan más de una detección. Afortunadamente, se dispone de resultados de un modelo en desarrollo

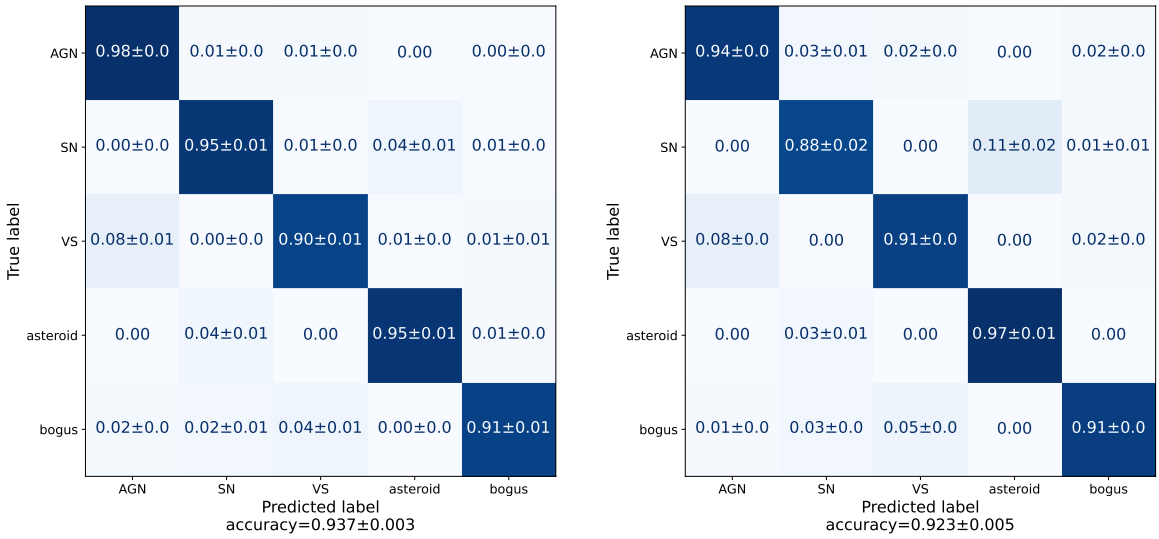
Tabla A.3: Clases asignadas a conjunto no etiquetado de 186,794 alertas de ZTF, por el *light curve classifier* (LCC) [1] y el stamp classifier (SC). El stamp classifier encuentra 78 % de las SN classificadas por el LCC, 85 % de los AGNs y 96 % de las VS. Los falsos positivos en la clase SN son un 9 % de AGNs, 6 % de VS, 4 % de asteroids y 3 % de bogus. Los falsos positivos de AGNs son 4 % de SN y 1 % de VS, y los falsos positivos de VS son solo 3 % de AGNs. La medida de *recall* en al última fila se mide en base a tomar las clasificaciones del LCC como etiquetas reales, es decir, mide que tan bien lo hace el stamp classifier, respecto al LCC. Tabla obtenida desde [2].

| SC \ LCC | SNIa | SNIbc | SNII | SLSN | AGN | Blazar | QSO | CV/Nova | YSO | DSCT | RRL | Ceph | LPV | EB | Otro-Periódico |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SN | 355 | 124 | 246 | 257 | 657 | 87 | 13 | 241 | 76 | 1 | 22 | 4 | 58 | 7 | 18 |
| AGN | 5 | 2 | 27 | 83 | 4057 | 1310 | 9553 | 592 | 309 | 227 | 1586 | 67 | 61 | 1133 | 1738 |
| VS | 10 | 7 | 22 | 38 | 393 | 623 | 691 | 2545 | 5023 | 5098 | 29635 | 9657 | 36478 | 56587 | 16539 |
| Asteroide | 19 | 8 | 6 | 17 | 1 | 4 | 1 | 38 | 10 | 0 | 7 | 1 | 107 | 8 | 1 |
| Bogus | 7 | 1 | 9 | 19 | 47 | 16 | 13 | 84 | 36 | 1 | 8 | 4 | 23 | 14 | 22 |
| **Recall** | 90 % | 87 % | 79 % | 62 % | 79 % | 64 % | 93 % | 73 % | 92 % | 96 % | 95 % | 99 % | 99 % | 98 % | 90 % |

por el equipo de ALeRCE, el *two-stamp-classifier* [5]. Este modelo recibe como entrada las dos primeras alertas de un objeto astronómico, siendo una simplificación del modelo propuesto en [30], donde se utilizan secuencias de imágenes de largo indefinido para clasificar objetos astronómicos. En [5] se compara el desempeño entre el two-stamp-classifier y el stamp classifier, sin regularización de entropía, sobre una versión de la base de datos ALeRCE-ZTF, que contiene dos alertas por cada uno de lo objetos que contiene, objetos que cuentan con las mismas 5 clases de etiquetas que la base de datos ALeRCE-ZTFv5 utilizada para entrenar el stamp classifier. Esto es posible ya que una cierta cantidad de bogus y asteroides generan más de una alerta. La generación de más de una alerta en objetos de tipo bogus se atribuye a fuentes muy luminosas que saturan las cámaras del telescopio, y para la clase asteroide son asteroides cuya velocidad tiene una dirección similar que una línea trazada entre el telescopio y el asteroide, por lo que aparecen en la misma posición entre dos observaciones cercanas en el tiempo.

Para el conjunto de test de la base de datos de ALeRCE-ZTF utilizada en [5], las Figuras A.1a y A.1b muestran las matrices de confusión del two-stamp-classifier y el stamp classifier, respectivamente. Se observa como el two-stamp-classifier logra una mejora de 1.4 % puntos de *accuracy* respecto al stamp classifier, y lo que es más importante, el two-stamp-classifier obtiene un *recall* de 95±1 % en la clase SN, mientras que el stamp classifier logra un *recall* de 88±2 % en SNe (mejora de 7 % puntos de *recall*), lo que resulta muy prometedor para la detección de SNe. Para más detalles de cómo se obtuvieron los resultados de la Figura A.1, y el funcionamiento del two-stamp-classifier, referirse a [5].

Se ha de destacar que las diferencias de desempeño obtenidas al comparar la matriz de confusión de la Figura 4.2 (Sección 4.1.2 de Resultados) y la Figura A.1b, obtenida de [5], se deben a que en cada caso se utilizan bases de datos diferentes, tanto para entrenar como para evaluar, pese a que el origen de los datos en ambos casos es ZTF, cada base de datos contiene diferentes objetos. Tambien se ha de tener en cuenta que la comparación entre el two-stamp-classifier y el stamp classifier no es del todo justa, ya que el two-stamp-classifier utiliza más información en su entrada.




(a) Two-stamp-classifier.

(b) Stamp classifier.

Figura A.1: Comparación de matrices de confusión promedio del two-stamp-classifier y el stamp classifier, evaluadas sobre el conjunto de test de una versión de la base de datos de ALeRCE-ZTF utilizada en [5]. (a) Two-stamp-classifier. (b) Stamp classifier. Figuras obtenidas de [5].

# Apéndice B

# Detección de anomalías tipo bogus

## B.1. Esquema de entrenamiento para grandes bases de datos

La implementación original de GeoTransform72 genera todas las transformaciones de los datos al comienzo del entrenamiento, lo que hace que el entrenamiento sea eficiente al tener todas las muestras disponibles en memoria RAM, pero el precio a pagar por esta eficiencia es ocupar grandes cantidades de RAM para almacenar el conjunto de datos auto-supervisado de entrenamiento. La cantidad de RAM utilizada aumenta de forma lineal con el número de transformaciones aplicadas, por lo cual resulta difícil entrenar con bases de datos con gran cantidad de muestras o muchas transformaciones.

Para solucionar el problema técnico descrito en el párrafo anterior, se opta por generar las transformaciones *en tiempo real* durante el entrenamiento, i.e., una vez que se extrae un *batch* de la base de datos original, para realizar una iteración de entrenamiento, a cada muestra del *batch* se le aplica una transformación $T_{\mathrm{i}} \in \mathcal{T}$ seleccionada de forma aleatoria y uniforme desde el conjunto de transformaciones $\mathcal{T}$. El problema de esta implementación es que aplicar las transformaciones *en tiempo real* resulta ser computacionalmente costoso, lo que genera un cuello de botella que aumenta los tiempos de entrenamiento hasta 10 veces. Este esquema de entrenamiento recibe el nombre *esquema para grandes bases de datos* en el resto del documento, y cada vez que sea utilizado para entrenar una versión de GeoTransform, se agregará el prefijo *large* al nombre del modelo, e.g. large-GeoTransform72.

En el esquema para grandes bases de datos se utiliza detención temprana, donde se espera la primera época de entrenamiento antes de comenzar a validar, para que el clasificador tenga la oportunidad de visualizar todas las muestras. Luego, se comienza a evaluar la entropía cruzada de validación cada 100 iteraciones con *patience* 3. La utilización de *patience* 3, se basa en permitir un horizonte más extenso para la exploración de un mejor clasificador en el valor de entropía cruzada de validación.

El esquema de entrenamiento *large* es una propuesta preliminar que necesita una exploración en mayor profundidad, donde se necesita ajustar adecuadamente sus hiperparámetros

de entrenamiento, como el *patience* o cantidad de iteraciones de entrenamiento.

A continuación se muestra una serie de pruebas realizadas al entrenar GeoTransform con el esquema de entrenamiento para grandes bases de datos. El modelo de GeoTransform utilizado corresponde a GeoTransform99 tras aplicarse el flujo de selección de transformaciones $M_1 \circ M_{2.1a} \circ M_3$-progresivo, i.e., el que obtiene los AUROC más altos en las Tablas 4.8 y 4.9, para small-HiTS y small-ALeRCE-ZTFv1, respectivamente. Es decir, se utiliza GeoTransform21 para small-HiTS y GeoTransform17 para small-ALeRCE-ZTFv1, que al ser entrenados con el esquema *large* son referidos como large-GeoTransform21 y large-GeoTransform17, respectivamente.

Para corroborar la correcta implementación del esquema de entrenamiento *large*, se evidencia si es que el desempeño de entrenar GeoTransform sobre las bases de datos reducidas es el mismo cuando se usa el esquema de entrenamiento con detención temprana y el esquema *large*. En detalle, se utilizan las bases de datos small-HiTS y small-ALeRCE-ZTFv1, para entrenar tanto GeoTransform21 y GeoTransform17 como large-GeoTransform21 y large-GeoTransform17. Los resultados de esta prueba se muestran en la Tabla B.1, donde se aprecia que para las dos bases de datos, ambos esquemas obtienen desempeños similares, lo que denota una correcta implementación del esquema de entrenamiento *large* y la capacidad del esquema para mantener el desempeño de GeoTransform. Al realizar esta prueba, también se mide el tiempo de entrenamiento de los modelos, que para el caso del esquema de entrenamiento con detención temprana es de 90 segundos en promedio, mientras que para el esquema *large* es de 15 minutos en promedio, i.e., el esquema *large* tarda 10 veces más en entrenar GeoTransform. En el futuro se espera realizar esta misma prueba sobre más modelos de GeoTransform, como GeoTransform72, para aumentar la certeza de la correcta implementación del modelo.

Tabla B.1: Corroboración la correcta implementación de esquema de entrenamiento *large*. Se compara el AUROC de entrenar GeoTransform sobre las bases de datos reducidas al utilizar el esquema de entrenamiento con detención temprana y el esquema *large*.

| Modelo | Esquema de entrenamiento | small-HiTS AUROC | small-ALeRCE-ZTFv1 AUROC |
|---|---|---|---|
| (small-HiTS) GeoTransform21<br>(small-ALeRCE-ZTFv1) GeoTransform17 | detención temprana | 99.42±0.09 | 94.14±0.39 |
| (small-HiTS) large-GeoTransform21<br>(small-ALeRCE-ZTFv1) large-GeoTransform17 | *large* | 99.44±0.07 | 94.26±0.71 |

Otra prueba que se realiza utilizando el esquema de entrenamiento *large* es entrenar sobre una base de datos con gran cantidad de muestras. Para esto se construye la base de datos que se denomina large-HiTS, compuesta por un conjunto de 70,000 inliers para entrenamiento, otro de 10,000 inliers para validación y un conjunto de test balanceado compuesto por 30,000 inliers y 30,000 bogus. Se tiene la precaución de que large-HiTS y small-HiTS no tengan muestras en común, lo que resulta fácil debido a que la base de datos de HiTS contiene más de 1 millón de muestras. Se entrena large-GeoTransform21 sobre large-HiTS, obteniéndose un AUROC de 98.38±0.55, donde cada modelo tarda 3 horas y 36 minutos en promedio, lo que resulta en un aumento considerable respecto a los 15 minutos que tarda en entrenarse sobre small-HiTS, lo que es esperable ya que large-HiTS es 10 veces más grande que small-HiTS.

Una comparación de entrenar large-GeoTransform21 con small-HiTS y large-HiTS puede verse en la Tabla B.2. Se aprecia como el desempeño sobre large-HiTS es menor al de small-HiTS, con una diferencia de 1.06 puntos de AUROC. Esto puede atribuirse a variadas razones como que los hiperparámetros de entrenamiento de large-GeoTransform21 no esten adecuadamente ajustados para large-HiTS, que existan diferencias en las distribuciones de cada uno de los conjuntos de small-HiTS y large-HiTS, siendo los conjuntos de small-HiTS no lo suficientemente representativos de large-HiTS, sumado al hecho de que la estrategia de selección de transformaciones que genera GeoTransform21 fue efectuada utilizando small-HiTS, y el conjunto de mejores transformaciones para large-HiTS puede ser otro.

Tabla B.2: Desempeño de entrenar GeoTransform bajo el esquema para grandes bases de datos (large-GeoTransform). El modelo siempre es entrenado sobre las versiones extendidas de la respectiva base de datos que se está evaluando. Se compara el desempeño de este esquema sobre las bases de datos reducidas y sus versiones extensas.

| Modelo | Esquema de entrenamiento | small-HiTS AUROC | large-HiTS AUROC |
|---|---|---|---|
| large-GeoTransform21 | large | 99.44±0.07 | 98.38±0.55 |

La estrategia de entrenamiento propuesta para grandes bases de datos (*large*), es un avance en la dirección correcta para poder utilizar GeoTransform en situaciones donde se cuenta con cantidades masivas de datos. Es fundamental su correcta implementación para que GeoTransform pueda aprovechar la información contenida en grandes cantidades de datos y generar modelos útiles para poner en producción y procesar el flujo de imágenes producidas por un estudio astronómico real. Se plantea como trabajo futuro evidenciar el desempeño de large-GeoTransform sobre bases de datos como ALeRCE-ZTFv7, además de evaluar el esquema *large* sobre otros modelos de GeoTransform, como GeoTransform72. Además, se necesitan más pruebas para ajustar los hiperparámetros de este esquemas de entrenamiento, lo que resulta complejo al no poderse usar conjuntos de validación que contengan anomalías.

## B.2. Método de selección de transformaciones 2.1.b - Matriz de distancia de Fréchet

El método $M_{2.1.b}$ se diseña de manera similar al método $M_{2.1.a}$, generando una matriz donde se miden similaridades entre pares de transformaciones para descartarlas. La única diferencia es que se mide una similitud basada en la distancia entre la distribución de datos presentes entre 2 pares de subconjuntos con transformaciones diferentes. Esta distancia se estima calculando la distancia de Frèchet (DF) [122] sobre las muestras del par de transformaciones comparadas. La DF se calcula al ajustar una distribución gaussiana multivariada sobre las imágenes de las muestras (cada pixel es una dimensión de la gaussiana) del par de transformaciones y calcular la distancia 2-Wasserstein entre ellas.

Cada elemento de la matriz que compara pares de transformaciones (matriz DF) con $M_{2.1.b}$, se construye a partir de la siguiente ecuación:

$$\left|\log \frac{DF(T_{\mathrm{i}}(\mathcal{X}), T_j(\mathcal{X}))}{DF(\mathcal{X}_1, \mathcal{X}_2)}\right| \tag{B.1}$$

Donde $T_{\mathrm{i}}$ y $T_j$ son el par de transformaciones que se está comparando, las cuales se aplican a todos los datos del conjunto de datos original $\mathcal{X}$, $DF(\cdot)$ es el cálculo de la distancia de Fréchet, $\mathcal{X}_1$ y $\mathcal{X}_2$ son subconjuntos de $\mathcal{X}$, cada uno compuesto por la mitad de los datos de $\mathcal{X}$. La idea detrás de la forma de calcular $M_{2.1.b}$, es normalizar la DF entre las transformaciones $T_{\mathrm{i}}$ y $T_j$, para tener una idea de qué valor significa que ambas sean similares o no, por lo cual la DF entre ellas se divide por la *auto*-DF, i.e., la DF de 2 subconjuntos de datos provenientes del mismo conjunto de datos $\mathcal{X}$.

Si es que el valor de la ecuación B.1 es menor a 1, se categoriza a las transformaciones como similares y una de ellas puede ser descartada como una transformación inútil de tipo $T_{inutil.2.1}$. La justificación de utilizar como valor de umbral 1 es que en este caso la DF entre las transformaciones esta a un orden de magnitud de distancia de la auto-DF. Finalmente el método $M_{2.1.b}$ queda definido por la siguiente definición.

**Definición B.2.1** (Método de selección $M_{2.1.b}$)**.** Al comparar dos transformaciones $T_{\mathrm{i}}$ y $T_j$, aplicadas sobre las muestras de $\mathcal{X}$, si se cumple:

$$\left|\log \frac{DF(T_{\mathrm{i}}(\mathcal{X}), T_j(\mathcal{X}))}{DF(\mathcal{X}_1, \mathcal{X}_2)}\right| < 1, \tag{B.2}$$

las transformaciones son equivalentes y una resulta redundante, i.e., se tiene una $T_{inutil.2.1}$.

Como caso particular, si es que una de las transformaciones es la identidad, entonces $\mathcal{X}$ es invariante a la otra transformación.

Para comparar los resultados de utilizar $M_{2.1.b}$ con el resto de los métodos de selección de transformaciones entrenados sobre small-HiTS y small-ALeRCE-ZTFv1, se agrega este método a la comparación de diferentes flujos de selección de transformaciones. En particular, se agregan los flujos $M_{2.1.b}$, $M_1 \circ M_{2.1.b}$, $M_1 \circ M_{2.1.b} \circ M_3$-progresivo y $M_1 \circ M_{2.1.b} \circ M_3$-regresivo a las Tablas 4.8 y 4.9. Con esto se generan las Tablas B.3 para small-HiTS y B.4 para small-ZTF.

Al observar los resultados de la Tabla B.3, donde se utiliza small-HiTS para entrenar, se aprecia que al seleccionar transformaciones para GeoTransform72 $M_{2.1.b}$ borra sólo algunas rotaciones y *flips*, dejando 65 transformaciones, mientras que para GeoTransform 99 sólo se borran 7 rotaciones. Esta es una inconsistencia del método $M_{2.1.b}$ ya que en el peor de los casos deberían ser eliminadas las mismas transformaciones. Esta inestabilidad ocurre ya que el método es sensible a las muestras escogidas al momento de calcular la *auto*-DF entre subconjuntos de muestras de los datos originales ($autoDF = DF(\mathcal{X}_1, \mathcal{X}_2)$), donde si estos conjuntos varían un poco la *auto*-DF cambia sustancialmente, lo que hace que el umbral escogido para categorizar un par de transformaciones como redundantes $M_{2.1.b} < 1$ sea poco confiable. Esta inestabilidad en el valor de la *auto*-DF se atribuye a que se está utilizando un conjunto muy pequeño de muestras para calcularla, ya que se trabaja sobre las bases de

Tabla B.3: AUROC para diferentes flujos de selección de transformaciones sobre base de datos small-HiTS, en conjuntos de transformaciones $\mathcal{T}_{juguete}$, GeoTransform72 y GeoTransform99. En negrita se indica el resultado con mayor AUROC promedio y menor desviación estándar de cada columna. Un *'No calculado'* significa que alguno de los métodos $M_3$ recibe más transformaciones que los límites fijados en este trabajo.

| Flujo de selección de transformaciones | 10 transformaciones con $\mathcal{T}_{juguete}$ AUROC (# transformaciones seleccionadas) | GeoTransform72 AUROC (# transformaciones seleccionadas) | GeoTransform99 AUROC (# transformaciones seleccionadas) |
|---|---|---|---|
| Ninguno | 99.18±0.11 (10) | 98.78±0.26 (72) | 99.12±0.04 (99) |
| $M_1$ | **99.18±0.10 (9)** | 98.78±0.26 (72) | 99.12±0.04 (99) |
| $M_{2.1.a}$ | 98.94±0.18 (7) | **99.16±0.13 (9)** | 99.20±0.06 (35) |
| $M_{2.1.b}$ | 99.11±0.14 (8) | 99.07±0.13 (65) | 99.37±0.10 (92) |
| $M_1 \circ M_{2.1.a}$ | 99.07±0.16 (6) | 99.16±0.13 (9) | 99.20±0.06 (35) |
| $M_1 \circ M_{2.1.b}$ | 99.14±0.13 (7) | 99.07±0.13 (65) | 99.37±0.10 (92) |
| $M_1 \circ M_{2.1.a} \circ$ $M_3$-progresivo | 99.07±0.16 (6) | 98.53±0.17 (4) | **99.42±0.09 (21)** |
| $M_1 \circ M_{2.1.a} \circ$ $M_3$-regresivo | 99.07±0.16 (6) | No calculado | No calculado |
| $M_1 \circ M_{2.1.b} \circ$ $M_3$-progresivo | 99.07±0.16 (6) | No calculado | No calculado |
| $M_1 \circ M_{2.1.b} \circ$ $M_3$-regresivo | 99.09±0.15 (6) | No calculado | No calculado |
| $M_3$-progresivo | 99.14±0.17 (6) | No calculado | No calculado |
| $M_3$-regresivo | 99.04±0.14 (7) | No calculado | No calculado |

datos reducidas. En el futuro se propone experimentar calculando $M_{2.1.b}$ sobre conjuntos de datos con más muestras, lo que aumenta la robustez de la estimación de las DF.

Al observar los resultados de $M_{2.1.b}$ sobre la base de datos small-ALeRCE-ZTFv1 en la Tabla B.4, la problemática anterior no se ve reflejada.

En resumen, el método de selección $M_{2.1.b}$, que busca el mismo tipo de transformaciones redundantes que $M_{2.1.a}$, tiene un comportamiento menos estable que $M_{2.1.a}$. Entregando resultados diferentes sobre los mismos conjuntos de transformaciones y sobre los mismos datos en oportunidades diferentes. Pese a que el trasfondo del método de comparar la distribución de probabilidad de los datos originales y sus transformaciones presenta fundamentos coherentes para lograr encontrar similitudes visuales entre bases de datos, su correcta implementación requiere más investigación.

Tabla B.4: AUROC para diferentes flujos de selección de transformaciones sobre base de datos small-ALeRCE-ZTFv1, en conjuntos de transformaciones $\mathcal{T}_{juguete}$, GeoTransform72 y GeoTransform99. En negrita se indica el resultado con mayor AUROC promedio y menor desviación estándar de cada columna. Un *'No calculado'* significa que alguno de los métodos $M_3$ recibe más transformaciones que los límites fijados en este trabajo.

| Flujo de selección de transformaciones | 10 transformaciones con $\mathcal{T}_{juguete}$ AUROC (# transformaciones seleccionadas) | GeoTransform72 AUROC (# transformaciones seleccionadas) | GeoTransform99 AUROC (# transformaciones seleccionadas) |
|---|---|---|---|
| Ninguno | 91.97±0.77 (10) | 87.33±1.24 (72) | 90.80±0.61 (99) |
| $M_1$ | **92.13±1.69 (9)** | 87.33±1.24 (72) | 90.80±0.61 (99) |
| $M_{2.1.a}$ | 90.99±1.15 (7) | 86.49±1.61 (9) | 91.39±0.76 (29) |
| $M_{2.1.b}$ | 89.54±2.10 (8) | 87.76±0.85 (65) | 91.19±0.41 (35) |
| $M_1 \circ M_{2.1.a}$ | 90.56±1.59 (6) | 86.49±1.61 (9) | 91.39±0.76 (29) |
| $M_1 \circ M_{2.1.b}$ | 89.98±1.47 (7) | 87.76±0.85 (65) | 91.19±0.41 (35) |
| $M_1 \circ M_{2.1.a} \circ$ $M_3$-progresivo | 76.27±5.85 (4) | **89.10±2.96** (6) | **94.14±0.39 (17)** |
| $M_1 \circ M_{2.1.a} \circ$ $M_3$-regresivo | 76.27±5.85 (4) | No calculado | No calculado |
| $M_1 \circ M_{2.1.b} \circ$ $M_3$-progresivo | 88.50±2.20 (6) | No calculado | No calculado |
| $M_1 \circ M_{2.1.b} \circ$ $M_3$-regresivo | 88.58±2.51 (6) | No calculado | No calculado |
| $M_3$-progresivo | 87.23±3.35 (7) | No calculado | No calculado |
| $M_3$-regresivo | 88.43±1.95 (8) | No calculado | No calculado |