Aprendizaje Generalizado de Cero Ejemplos: Revisión de Estado del Arte y Estudio Preliminar de la Influencia del Preprocesamiento

Juan José Herrera Aranda*, Francisco Herrera*, Isaac Triguero*,†
*Instituto Andaluz Interuniversitario en Data Science and Computational Intelligence,
Departamento de Ciencias de la Computación e Inteligencia Artificial, Universidad de Granada
†School of Computer Science, University of Nottingham, Nottingham, NG8 1BB, United Kingdom
Email: {juanjoha, herrera, isaaktriguero}@ugr.es

***Resumen*—La inteligencia artificial de propósito general es uno de los objetivos más ambiciosos en la actualidad. Para alcanzarlo se requiere de nuevas técnicas, especialmente en un contexto de mundo abierto, en el que nuevas tareas y cambios en el entorno pueden aparecer, necesitando más dinamismo de estos sistemas. Entre ellas destaca el Aprendizaje de Cero Ejemplos, siendo útil para muchas áreas en las que los conjuntos de entrenamiento y prueba tienen etiquetas disjuntas, aunque existe una versión más general en la que la intersección en el conjunto de prueba ya no es disjunta, presentando nuevos retos dada su complejidad. Esta contribución propone una nueva perspectiva organizativa de esta rama dada la reciente aparición de propuestas basadas en grandes modelos del lenguaje, las cuales abren nuevas líneas de investigación. Una de ellas es el uso de datos tabulares, los cuales han pasado desapercibidos hasta el momento. Además, se explora una nueva línea de investigación en la que el foco no está tanto en la técnica de aprendizaje, como la mayoría de contribuciones existentes en la literatura, sino en el preprocesamiento del espacio semántico mediante técnicas avanzadas para estudiar la influencia en el rendimiento de los modelos. Para ello se realiza un experimento preliminar que evidencia las posibilidades de mejora basadas en preprocesamiento.**

***Palabras clave*—Inteligencia Artificial de Propósito General, Reconocimiento de Conjuntos Abiertos, Aprendizaje de Cero Ejemplos, Aprendizaje de Cero Ejemplos Generalizado, Grandes Modelos del Lenguaje**

I. Introducción

Hasta la fecha, la mayoría de las aplicaciones de Inteligencia Artificial (IA) se diseñan para propósitos específicos, destacando en la resolución de tareas definidas, pero dependiendo usualmente de grandes cantidades de datos para un rendimiento efectivo. A medida que la IA continúa evolucionando, surge una creciente demanda de aplicaciones que requieren Sistemas de Inteligencia Artificial de Propósito General (*General-Purpose Artificial Intelligence*, GPAIS) [1], en particular, de aquellas que son de mundo abierto, en donde los entornos son dinámicos con la aparición de nuevos tipos de datos y tareas a resolver.

En contraste con la suposición de los GPAIS de mundo cerrado, prevalente en escenarios donde no se puede generalizar más allá de los datos de entrenamiento, las aplicaciones reales operan en entornos abiertos. En este contexto, el Reconocimiento de Conjunto Abierto (*Open Set Recognition*, OSR) juega un papel crucial [2] por ser un paradigma que permite tratar con situaciones desconocidas en la fase de prueba. En el campo del Aprendizaje Automático (*Machine Learning*, ML), el OSR engloba métodos que tratan de manejar escenarios desconocidos que no se cubrieron durante la fase de entrenamiento de un modelo. Una aplicación directa de lo anterior es la clasificación de enfermedades de un paciente. Aquellas detectadas como desconocidas alertan a los expertos de la necesidad de un diagnóstico más concreto.

Otro paradigma significativo que aborda situaciones desconocidas en la fase de prueba y que ha cobrado importancia recientemente es el Aprendizaje de Cero Ejemplos (*Zero-Shot Learning*, ZSL). Mientras que el OSR es capaz de detectar únicamente aquellas situaciones con las que el modelo no ha sido entrenado, ZSL va más allá, ya que las clasifica. Para ello, sin embargo, necesita de información semántica, esto es, información auxiliar con descripciones (p.ej. atributos) del mundo en el que opera el sistema. La hipótesis en ZSL es que las etiquetas de las clases de entrenamiento (clases vistas) son disjuntas de las etiquetas de las clases de prueba (clases no vistas). Sin embargo, en la práctica, es común encontrar ejemplos tanto de las clases vistas como de las no vistas en la fase de *test* [3]. Este paso más allá del ZSL se conoce como Aprendizaje de Cero Ejemplos Generalizado (*Generalised Zero-Shot Learning*, GZSL). La coexistencia entre estos dos tipos de clases en la fase de prueba implica nuevos desafíos debido a la tendencia a predecir lo ya conocido. En [3], se realiza una extensa revisión y categorización de la literatura. No obstante, esta área avanza a tal velocidad que las últimas tendencias basadas en Grandes Modelos de Lenguaje (*Large Language Models*, LLMs) [4] no están incluidas.

En esta contribución, se explora las propuestas recientes de la literatura basadas en LLMs. Estos enfoques no han sido cubiertos en otras revisiones bibliográficas y actualmente están remodelando el campo, abriendo un nuevo abanico de oportunidades. Por esta razón, es esencial documentar estos nuevos enfoques y proponer una nueva perspectiva organizativa además de una categorización en lo que respecta espacios semánticos. También, se discuten otros aspectos críticos que a menudo se han pasado por alto, como por ejemplo, la

posibilidad de que un LLM conozca de antemano el conjunto de datos sobre al que se va a aplicar. En lo que respecto al tipo de datos usados, mientras que la mayoría de los estudios se centran en imágenes, texto y vídeos, los datos tabulares han pasado desapercibidos. Sin embargo, la aparición de estas nuevas propuestas basadas en LLMs han abierto nuevas posibilidades y desafíos que serán analizados.

Por otra parte, la mayoría de las contribuciones actuales se centran en la creación de modelos, y pocos estudios se han centrado en la importancia del preprocesamiento para GZSL. Si bien algunas propuestas recientes tratan esta temática, no ha sido el objetivo principal, como por ejemplo en [5] que se centra en la fusión de datos. En este trabajo, se estudia la posibilidad de preprocesar la información semántica para mejorar el rendimiento. Recientemente, en [6], se analiza la importancia de los atributos del espacio semántico, a posteriori, sin realizar preprocesamiento antes de entrenar un modelo. En esta contribución se explora la importancia del preprocesamiento, mediante un experimento inicial que usa el algoritmo de selección de características *Recursive Feature Elimination* (RFE) [7] en el espacio semántico, para posteriormente integrarlo con la propuesta dada por [8] basada en autocodificadores donde el espacio latente es dicho espacio semántico.

Este trabajo se organiza como sigue. La Sección II habla sobre el trabajo relacionado con las revisiones bibliográficas actuales, subrayando la necesidad de actualizarlas. La Sección III introduce la notación y el contexto del ZSL/GZSL, además provee una nueva categorización de los espacios semánticos para dar cabida a literatura reciente. Seguidamente, la Sección IV plantea un experimento preliminar para mostrar el potencial que tiene el preprocesamiento del espacio semántico. Finalmente, la Sección V concluye este trabajo discutiendo los retos a los que se enfrentan ZSL/GZSL, y trabajo futuro.

II. Trabajo Relacionado

Esta sección proporciona una revisión de los estudios bibliográficos tanto en ZSL como en GZSL. Se identifican áreas en las que no se ha trabajado en la literatura. Además, se examinan las diversas categorizaciones existentes, proporcionando una visión más clara de los enfoques utilizados. Por último, se comentará la necesidad de actualizarlos debido al avance experimentado en los últimos años.

En el contexto de clasificación en presencia de clases desconocidas, han surgido muchas metodologías que tratan de abordar esto de una manera u otra, como podría ser el clasificador de una clase (*One-Class Classification*, OCC) [9], la detección de fuera de la distribución (*Out-of-Distribution*, OOD) [10], OSR [11] y ZSL [12] entre otros, dando lugar a cierta confusión debido a la falta de consenso en la comunidad que delimite similitudes y diferencias entre ellas.

Con el fin de aclarar bajo qué configuraciones pueden operar algunos de estos paradigmas, en [2] se distingue entre cuatro categorías básicas de clases, recalcando la diferencia entre OSR y ZSL. Sin embargo, dicha diferencia puede resultar un poco confusa atendiendo a [13], donde se proponen una serie de configuraciones bajo las cuales puede operar ZSL. Estas configuraciones son inductivas (la información no está disponible) o transductivas (la información está disponible) tanto en instancias (refiriéndose a las instancias de *test*, ya que las de entrenamiento están siempre disponibles) como de clase (refiriéndose al espacio semántico del conjunto de *test*) en lo que respecta a la fase de entrenamiento.

La distinción principal no radica tanto en aspectos técnicos, sino en el propósito previsto. Los modelos de OSR se han centrado en discernir entre aquellas situaciones que ya se conocen y aquellas que no, sin profundizar en estas últimas. Mientras que los modelos ZSL se usan para hacer distinciones dentro del conjunto de clases que no se conocen. No obstante, en GZSL se pueden aplicar una combinación de técnicas de OSR o detección de elementos fuera de la distribución con ZSL para abordar estos problemas [3].

En la literatura, se pueden encontrar varios estudios bibliográficos tanto de ZSL como de GZSL [3], [13], [14]. En [13], se distinguen dos tipos principales de métodos: **basados en clasificadores** y **basados en instancias**. Los primeros se centran en aprender un clasificador que sea capaz de identificar elementos de clases no vistas. Los segundos se basan en asignar etiquetas a los elementos de las clases no vistas para posteriormente usarlas en el aprendizaje. Asimismo, se clasifican los espacios semánticos en otros dos tipos: **espacios semánticos diseñados** y **aprendidos**. La diferencia fundamental radica en que los primeros son diseñados por humanos, mientras que los segundos se obtienen a partir de la salida de un modelo de aprendizaje automático.

A diferencia de otras revisiones, [3] profundiza en las diferencias de GZSL. En particular, el GZSL se aborda fundamentalmente desde una perspectiva inductiva de instancia, dado que según los autores, es más realista que una perspectiva transductiva de instancia, ya que bajo la primera no se tiene información de los elementos del conjunto de *test*, mientras que en la otra sí. La categorización propuesta se centra en cómo transferir conocimiento entre clases vistas y no vistas, y en cómo hacer que el modelo aprenda a reconocer imágenes de ambas clases sin tener información de las etiquetas de las clases no vistas. De acuerdo a esto, se distingue entre **métodos basados en espacios embebidos** y **métodos generativos**.

- **Métodos basados en espacios embebidos**: Aprenden un espacio embebido para asociar las características de bajo nivel de las clases vistas con sus correspondientes vectores semánticos. Posteriormente, se aprende una función proyectora para reconocer nuevas clases, midiendo la similitud entre los vectores semánticos verdaderos y los predichos de las muestras en el espacio de embebimiento. Estos métodos a su vez se categorizan en: basados en detección de elementos fuera de la distribución, basados en grafos, basados en meta aprendizaje, basados en atención, aprendizaje bidireccional, basados en autocodificadores.
- **Métodos generativos**: El modelo trata de generar imágenes o representaciones visuales de las características no vistas basadas en los elementos de las clases vistas, convirtiendo así el GZSL en un problema de

clasificación supervisada convencional. Estos métodos no pueden ser puramente inductivos, ya que al menos se necesita información semántica de las clases no vistas para poder operar. Estos métodos se dividen en Redes Adversarias Generativas, Autocodificadores Variacionales y sus combinaciones.

El intenso avance en los últimos años ha impulsado el desarrollo de nuevos métodos, particularmente aquellos basados en LLMs y *Transformers*, que no están recogidos en las categorías anteriores. La siguiente sección ahonda en estos métodos.

III. Revisión del estado del arte en (G)ZSL

En esta sección se presenta una nueva estructura organizativa incorporando a los LLMs a la categorización dada por [3]. La Subsección III-A define formalmente el problema de ZSL y GZSL. La Subsección III-B expone una nueva categorización para espacios semánticos y se comentan las implicaciones de usar los LLMs. Finalmente, la Subsección III-C explora la principal línea de investigación que abren las LLMs en lo que respecta al tipo de datos para problemas de ZSL y GZSL.

A. Preliminares y notación

Dados $N_s, N_u \in \mathbb{N}$, se definen los conjuntos, $\mathcal{S} \triangleq \{c_i^s\}_{i=1}^{N_s}$ y $\mathcal{U} \triangleq \{c_i^u\}_{i=1}^{N_u}$ como las clases vistas y no vistas, respectivamente, cumpliendo además que $\mathcal{S} \cap \mathcal{U} = \emptyset$. El espacio de características será un espacio d-dimensional de números reales, $\mathcal{X} \triangleq \mathbb{R}^d$, del cual se extraerán un conjunto de muestras $N_{tr}, N_{te} \in \mathbb{N}$ asociadas a $\mathcal{S}$ y $\mathcal{U}$, conformando los conjuntos de entrenamiento y *test*, respectivamente,

$$X^{tr} \triangleq \{x^T \in \mathcal{X} \ : \ c(x) \in \mathcal{S}\} \tag{1}$$
$$X^{te} \triangleq \{x^T \in \mathcal{X} \ : \ c(x) \in \mathcal{U}\} \tag{2}$$

donde $c(\cdot) : \mathcal{X} \rightarrow \mathcal{S} \cup \mathcal{U}$ es la función que dada una instancia devuelve su etiqueta correspondiente. En caso de estar bajo el paradigma de GZSL el conjunto de *test* incluirá elementos tanto de $\mathcal{S}$ como de $\mathcal{U}$.

El espacio semántico se definirá también como un espacio de números reales m-dimensional, $\mathcal{T} \triangleq \mathbb{R}^m$ y sus elementos serán llamados prototipos. Cada clase tendrá asociada, mediante la función proyectora $\Pi(\cdot) : \mathcal{S} \cup \mathcal{U} \rightarrow \mathcal{T}$, un único prototipo. Por lo que se define un espacio semántico específico para el conjunto de clases vistas, $\mathcal{T}^S \triangleq \{t_i^s\}_{i=1}^{N_s}$ y otro para el conjunto de clases no vistas, $\mathcal{T}^U = \{t_i^u\}_{i=1}^{N_u}$.

De cara a la explicación del método de GZSL que se usa experimentación, se definen también los conjuntos

$$\mathcal{T}^{tr} = \{(\Pi \circ c)(x) : x \in X^{tr}\} \tag{3}$$
$$\mathcal{T}^{te} = \{(\Pi \circ c)(x) : x \in X^{te}\} \tag{4}$$

que representan los prototipos asociados a las instancias de entrenamiento y *test* respectivamente.

Dado el conjunto de entrenamiento $(X^{tr}, c(X^{tr}))$ y su respectiva información semántica asociada $\mathcal{T}^{tr}$, la tarea de ZSL consiste en aprender un clasificador $f^u(\cdot) : X^{te} \rightarrow \mathcal{U}$, que asigna una etiqueta de la clase no vista a cada elemento del conjunto de *test*. Bajo el contexto de GZSL, el clasificador tendrá un dominio y codominio más amplio, $f(\cdot) : \mathcal{X} \rightarrow \mathcal{S} \cup \mathcal{U}$. De hecho, se puede entender ZSL como un caso particular de GZSL en donde $f^u = f_{|X^{te} \rightarrow \mathcal{U}}$. Por último, comentar que dependiendo de si se está en un contexto inductivo o transductivo, en la fase de entrenamiento se tendría también acceso a $\mathcal{T}^{te}$ (contexto inductivo de instancia y transductivo semántico) o a $X^{te}, \mathcal{T}^{te}$ (puramente transductivo).

B. Nueva categorización de Espacios Semánticos e Implicaciones al usar LLMs

Como se indicó anteriormente, hasta el momento se han considerado dos tipos de espacios semánticos: diseñados y aprendidos. Partiendo de la categorización de [13], se plantea una extensión que incorpora las nuevas contribuciones basadas en LLMs, añadiendo así un tercer tipo de espacio semántico. Entre los estudios más recientes destaca [6], donde se compara su propuesta con Chat-GPT3 [15] para tareas de ZSL. Otras aproximaciones, como [5], aplican LLMs para consolidar varias bases de datos en una única y posteriormente aprender un modelo que les sirva en tareas de aprendizaje de pocos datos (conocido en inglés como *few-shot learning*). De hecho, en [16] ya se propone un estudio para profundizar más en el conocimiento oculto dentro de los LLMs.

Para dar cabida a estos trabajos, se propone distinguir entre espacios semánticos explícitos e implícitos (Ver Figura 1).

- **Espacios semánticos explícitos**: Son aquellos a los que el ingeniero puede acceder y modificar. Esta categoría incluye los espacios tanto diseñados como aprendidos junto con todos sus subtipos tratados en [13]
- **Espacios semánticos implícitos**: Vienen incorporados por los LLMs y se caracterizan por no tener acceso ni poder ser modificados por parte del desarrollador del modelo de ZSL/GZLS, ya que actúan como cajas negras. Dado que los LLMs se entrenan con enormes cantidades de datos, se especula que en ZSL/GZSL, tienen un espacio semántico más rico que cualquier espacio explícito, pudiendo mejorar considerablemente los resultados.

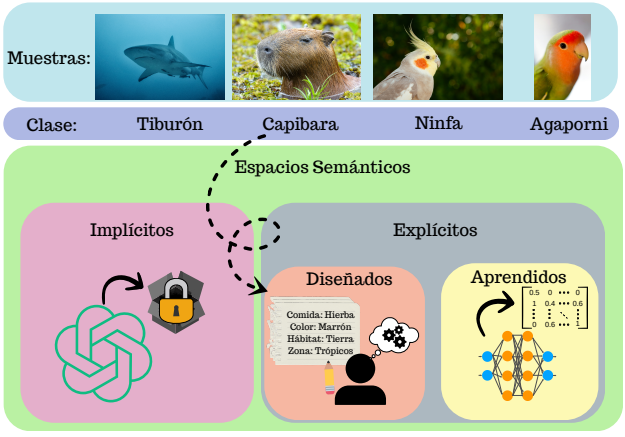

Fig. 1. Ampliación de la categorización propuesta en [13] de los espacios semánticos. La aportación de este trabajo corresponde a la inclusión de los espacios implícitos dentro de la categorización anterior.

A pesar del rendimiento que demuestran las LLMs para tareas de ZSL, estas puede acarrear varios problemas. Las bases de datos usadas por la comunidad científica para ZSL son de libre acceso y bastante conocidas. Por lo que especulamos que los LLMs pueden llegar a conocer los datos que se usan en *test*, provocando así una contaminación en los resultados. En estudios recientes en GZSL como [5], [6] no tienen en cuenta esta posible fuga de información.

C. GZSL para Datos Tabulares

La mayoría de los estudios en ZSL/GZSL se centran en imágenes [17] y vídeos [18], pero no en datos tabulares. La razón principal es que las imágenes cuentan información local-espacial, lo que resulta en un espacio de características más flexible y rico para entrenar modelos en comparación con los datos tabulares. Además, la creación de espacios semánticos para imágenes o vídeos ha sido una tarea natural para mejorar la expresividad de los mismos, lo que ha favorecido el uso de este tipo de datos. No obstante, están surgiendo enfoques que trabajan con datos tabulares, gracias al potencial de técnicas basadas en *Transformers* [19], [20] y LLMs [5], [6].

Para utilizar un LLM para datos tabulares, estos han de serializarse en una representación de texto natural. Es importante destacar el rendimiento de los LLMs es muy sensible a la manera en la que se presenta la entrada en formato de lenguaje natural [21]. En [6], se evalúan varios métodos que generan texto natural para crear entradas que se acerquen más a la distribución de entrenamiento del LLM. Utilizan un modelo LLM preentrenado y aplican un ajuste fino para realizar tareas de pocos y cero ejemplos en conjuntos de datos tabulares bien conocidos. Los resultados en la tarea ZSL fueron bastante satisfactorios, superando incluso los resultados de ChatGPT3.

IV. Preprocesamiento del espacio semántico: un experimento inicial

Gran parte de los trabajos en ZSL/GZSL se centran en las técnicas de aprendizaje. Sin embargo, hay poca literatura que se relacione con el preprocesamiento (instancias o espacio semántico). Este trabajo se centra en el espacio semántico (explícito), ya que hasta el momento se ha asumido que los espacios diseñados o aprendidos son óptimos para la creación de un modelo. En algunas contribuciones como [6] hacen un estudio sobre la importancia de los atributos del espacio semántico, pero no lo aplican como una tarea de preprocesamiento.

Esta experimentación preliminar tiene el propósito de mostrar la necesidad de aplicar un preprocesamiento al espacio semántico para mejorar los resultados de los modelos. Para ello se ha adaptado la propuesta dada en [8] detallada en la Subsección IV-A y aplicada sobre el dataset *Caltech-UCSD Birds-200-2011* (CUB200-2011) [22], explicado en la Subsección IV-B. Posteriormente, en la Subsección IV-C se detallará el preprocesamiento aplicado en el espacio semántico. Finalmente, en la Subsección IV-D se analizarán los resultados.

A. Autocodificador Semántico

Para este experimento, se ha elegido un Autocodificador Semántico (*Semantic Autoencoder*, SAE) [8] que pertenece a la categoría **métodos basados en espacios embebidos**, como método ampliamente usado en el área. Está basado en el paradigma codificador-decodificador. Por una parte, el codificador proyecta las características de X^{tr} en el espacio semántico $\mathcal{T}^{tr}$ por medio de una transformación lineal cuyos parámetros vienen dado por una matriz $W \in \mathbb{R}^{m \times d}$. Por otra parte, el decodificador, partiendo de $\mathcal{T}^{tr}$, intenta reconstruir las características lo más fielmente a las originales, produciendo un nuevo espacio de características $\widehat{X}^{tr}$. Se incluye una restricción adicional: la matriz dada por la transformación lineal del decodificador ha de ser la matriz traspuesta de la transformada que aplicada el codificador. La formulación del problema es la que sigue:

$$\min_{W,W^t} \|X - W^T W X^T\|_F^2 \quad s.t. \quad WX^T = \mathcal{T}^{tr} \tag{5}$$

Dicha ecuación se resuelve relajando la restricción dura $WX^T = \mathcal{T}^{tr}$ por una restricción más suave, reformulando lo anterior como:

$$\min_{W,W^t} \|X - W^T W X^T\|_F^2 + \lambda \|WX^T - \mathcal{T}^{tr}\|_F^2 \tag{6}$$

donde λ es un coeficiente que controla la importancia del segundo término repercutiendo en mayor o menor cantidad a la hora de calcular los pesos. Es importante notar que un mayor valor de λ dará mucho más peso al segundo término, obligando a que $WX^T \approx \mathcal{T}^{tr}$, mientras que un valor más bajo, relajará la restricción en favor del segundo. Un buen ajuste de λ será crucial.

Tras desarrollar (6), se llega la siguiente formulación:

$$AW + WB = C \tag{7}$$

donde $A = \mathcal{T}^{tr}(\mathcal{T}^{tr})^T, B = \lambda X^T X$ y $C = (1+\lambda)\mathcal{T}^{tr}X$. Dicha ecuación se conoce como la ecuación de Sylvester y puede ser resuelta de manera eficiente. La complejidad de (7) es $\mathcal{O}(k^3)$, dependiendo cúbicamente del número de características de X. Existen dos maneras de evaluar en este método, según se aplique o bien la transformada del codificador o bien la del decodificador. Se explicará el primer método por ser el que se ha usado dado que en los experimentos obtiene mejores resultados.

Al evaluar, se parte de un ejemplo de *test* $x_i \in X^{te}$ y se obtiene su proyección $\hat{t_i} = Wx_i$. Para clasificarlo, se calcula la distancia entre $\hat{t_i}$ y todas las representaciones en $\mathcal{T}^U$, tomando como clase el prototipo más cercano:

$$\phi(x_i) = \arg\min_j D(\hat{t}_i, t_j^u) \tag{8}$$

B. Dataset: CUB200-2011

El conjunto de datos empleado, CUB200-2011 [22], es una versión ampliada de CUB200, duplicando el número de imágenes por categoría y añadiendo nuevas anotaciones de

localización de partes sobre el conjunto original. Se tienen en total 11.788 imágenes pertenecientes a 200 especies de aves y 312 atributos semánticos para cada una de las clases.

En lugar de partir de las imágenes, se utilizan las características obtenidas con la arquitectura ResNet101 implementada en [14]. De esta manera, cada imagen tendrá asociada un vector de 2048 características. Por otra parte, también se ha seguido la misma partición que proponen estos autores para seleccionar los conjuntos de entrenamiento, X^{tr}, y *test*, X^{te}, y para determinar cuáles serán las clases vistas, $\mathcal{S}$, y no vistas, $\mathcal{U}$. Se seleccionan 150 clases como vistas $\mathcal{S}$ y 50 como no vistas, $\mathcal{U}$. Se tienen 7057 instancias de entrenamiento, todas con etiquetas pertenecientes a $\mathcal{S}$, y 4731 instancias para el *test*, de las cuales 1764 tienen etiquetas pertenecientes a $\mathcal{S}$ y 2967 etiquetas pertenecientes a $\mathcal{U}$. Además, se ha usado como espacio semántico los 312 atributos para cada unas de las 200 clases. Así, el conjunto de entrenamiento tendrá la forma $(X^{tr}, \mathcal{T}^{tr}, \mathcal{S})$ y el de *test* $(X^{te}, \mathcal{T}^{te}, \mathcal{S} \cup \mathcal{U})$. Siguiendo el enfoque más común de la literatura, el espacio semántico será usado en su versión continua, en la que cada atributo tiene un número real entre 0 y 100, asignado por una persona, que considera cuánto del atributo está presente en la clase.

C. Preprocesamiento

Las características de los datos proporcionadas por ResNet101 son normalizadas, restando por su media y dividiendo por la desviación típica de los conjuntos de entrenamiento y *test*, análogo para el espacio semántico. Después, se aplica una selección de características al espacio semántico por medio del RFE [7] cuyo estimador ha sido una una máquina de soporte vectorial con un *kernel* lineal.

El conjunto de entrenamiento usado por el algoritmo RFE ha sido $\mathcal{T}^{tr}$, que corresponde a los atributos del espacio semántico asociados a cada instancia del conjunto de entrenamiento. Para evaluar el rendimiento del clasificador se ha calculado la tasa de acierto. Se han aplicado en total 62 ejecuciones del algoritmo, seleccionando desde 10 hasta 312 características del espacio semántico con un incremento de 5 en 5 y seleccionando en cada iteración las mejores. Tras finalizar todas las iteraciones, el algoritmo devuelve aquella selección de atributos que mayor porcentaje de acierto han tenido al evaluar el modelo sobre el conjunto de validación, extraído del propio conjunto de entrenamiento.

D. Resultados

En este experimento, el mejor resultado se obtiene seleccionando 175 atributos de los 312, una reducción bastante significativa. Tras ello, se evalúa en el conjunto de *test* el nuevo espacio semántico y se compara los resultados con los del original en la Tabla I. Para entender cómo influye el preprocesamiento del espacio semántico en el rendimiento del modelo, se distingue en fase de *test* entre instancias pertenecientes a $\mathcal{S}$ y a $\mathcal{U}$, lo cual no se sabe en tiempo de inferencia. En vista de los resultados dados por la Tabla I, se aprecia una mejora de la tasa de acierto aplicando el espacio semántico obtenido por el conjunto de atributos reducido para las instancias de *test* correspondientes al conjunto $\mathcal{S}$, pero por otra, empeoran para $\mathcal{U}$. Este resultado tiene mucho sentido, ya que la selección de atributos ha estado guiada por el resultado en $\mathcal{S}$ sin tener en cuenta ninguna información de $\mathcal{U}$. Se plantean así las siguientes cuestiones que requerirán trabajo futuro:

- ¿Existe un preprocesamiento que mejore los resultados tanto para las clases vistas como las no vistas?
- ¿Puede darse el caso de que haya atributos que no tengan relevancia en el conjunto $\mathcal{S}$ pero que sí la tengan en $\mathcal{U}$?
- Suponiendo que existiera dos preprocesamientos, uno que mejorase los resultados de $\mathcal{U}$ y otro que mejorarse los de $\mathcal{S}$ ¿Se podría crear un modelo que en tiempo de inferencia sepa si la instancia viene de $\mathcal{S}$ o $\mathcal{U}$ para determinar que tipo de preprocesamiento aplicar?

Para responder la primera pregunta, supóngase que existe un oráculo que tiene acceso a la información del *test* y proporciona los resultados de aplicarlo a cada uno de los 62 espacios semánticos generados. Los resultados más significativos se recogen en la Tabla II. Se aprecia que el espacio semántico con 245 atributos mejora al espacio original en los dos tipos de clases, mostrando así lo pretendido. En Fig. 2 se puede ver la evolución del rendimiento de los distintos espacios semánticos al aplicarle ambos conjuntos de *test*. Ambos poseen una forma similar, la gráfica adquiere una tendencia creciente, hasta un umbral a partir del cual permanecen oscilando, sugiriendo que añadir más atributos no hace que haya mejoras significativas. Es importante subrayar que pese a que existe una solución, no se puede llegar a ella, ya que para ello se ha necesitado conocimiento del conjunto de *test*, lo cual no es realista.

Con respecto a la segunda pregunta, la respuesta depende del conjunto de datos. En aquellos que son de *grano-fino*, como este, es posible que los atributos que tengan relevancia para las instancias que pertenecen a $\mathcal{S}$, también la tengan para las de $\mathcal{U}$, ya que es un conjunto de datos con distintas especies de aves, esto es, el conjunto de datos tiene una semántica homogénea. De hecho, en Fig. 2 se observa que las curvas son muy parecidos en cuanto a la forma, siendo un indicativo de lo dicho. En cambio, para otros conjuntos de datos con semánticas heterogéneas, como aquellos que tengan elementos sin relación entre sí, podría darse el caso opuesto.

En respuesta a la tercera pregunta realizada, existen soluciones que aplican técnicas de OOD, considerando a los elementos dentro de la distribución como las instancias con etiquetas pertenecientes a $\mathcal{S}$ y los que están fuera como las instancias con etiquetas pertenecientes a $\mathcal{U}$ [23]. De esta manera, se podría saber en tiempo de inferencia qué preprocesamiento aplicar al espacio semántico de las instancias de cada tipo de clase en caso de no tener uno único.

V. Conclusiones

En este trabajo, se ha realizado una revisión del estado del arte en Clasificación de Cero Ejemplos Generalizado. A diferencia de trabajos anteriores, se han analizado las últimas propuestas basadas en modelos de lenguaje, junto con la posibilidad de aplicarlos en conjuntos de datos tabulares. También

TABLA I
RESULTADOS OBTENIDOS TRAS LA EXPERIMENTACIÓN.

Atributos	$\mathcal{S}$	$\mathcal{U}$
175	62,47	40,68
312	60,71	42,33

TABLA II
ALGUNOS DE LOS RESULTADOS OBTENIDOS POR EL ORÁCULO

Atributos	$\mathcal{S}$	$\mathcal{U}$
245	61,39	44,49
312	60,71	42,33

se ha explicado que es posible que estos modelos conozcan de antemano los datos de *test*, siendo uno de sus principales inconvenientes. Además, se ha propuesto una nueva manera de categorizar los espacios semánticos. Por último, se ha realizado un estudio preliminar sobre la influencia del preprocesamiento del espacio semántico, identificando que es posible, pero de momento no alcanzable, abriendo así una nueva oportunidad de investigación.

AGRADECIMIENTOS

Cabe destacar que esta iniciativa se realiza en el marco de los fondos del Plan de Recuperación, Transformación y Resiliencia, financiados por la Unión Europea (Next Generation).

REFERENCIAS

[1] I. Triguero, D. Molina, J. Poyatos, J. Del Ser, and F. Herrera, "General purpose artificial intelligence systems (gpais): Properties, definition, taxonomy, societal implications and responsible governance," *Information Fusion*, vol. 103, p. 102135, 2024.

[2] C. Geng, S.-j. Huang, and S. Chen, "Recent advances in open set recognition: A survey," *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 10, pp. 3614–3631, 2020.

[3] F. Pourpanah, M. Abdar, Y. Luo, X. Zhou, and et al, "A review of generalized zero-shot learning methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 4, pp. 4051–4070, 2023.

[4] W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, and et al, "A survey of large language models," *arXiv preprint arXiv:2303.18223*, 2023.

[5] Z. Wang, C. Gao, C. Xiao, and J. Sun, "Meditab: Scaling medical tabular data predictors via data consolidation, enrichment, and refinement," 2023.

[6] S. Hegselmann, A. Buendia, H. Lang, M. Agrawal, X. Jiang, and D. Sontag, "TabLLM: Few-shot classification of tabular data with large language models," in *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, vol. 206. PMLR, 2023, pp. 5549–5581.

[7] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene selection for cancer classification using support vector machines," *Machine learning*, vol. 46, pp. 389–422, 2002.

[8] E. Kodirov, T. Xiang, and S. Gong, "Semantic autoencoder for zero-shot learning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 3174–3183.

[9] B. Krawczyk, M. Woźniak, and B. Cyganek, "Clustering-based ensembles for one-class classification," *Information Sciences*, vol. 264, pp. 182–195, 2014, serious Games.

[10] J. Liu, Z. Shen, Y. He, X. Zhang, R. Xu, H. Yu, and P. Cui, "Towards out-of-distribution generalization: A survey," *arXiv preprint arXiv:2108.13624*, 2021.

[11] A. Mahdavi and M. Carvalho, "A survey on open set recognition," in *2021 IEEE Fourth International Conference on Artificial Intelligence and Knowledge Engineering (AIKE)*. Los Alamitos, CA, USA: IEEE Computer Society, 2021, pp. 37–44.

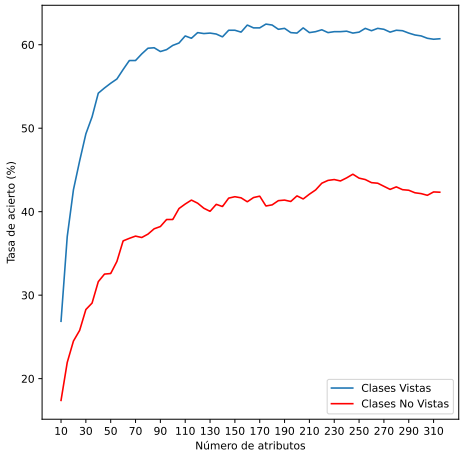

Fig. 2. Evolución de la tasa de acierto, dada por el oráculo, con respecto el número de características seleccionadas en el espacio de atributos. La gráfica azul y roja corresponden a la evaluación sobre el conjunto de clases vistas y no vistas respectivamente, del conjunto de test.

[12] J. Liu, C. Shi, D. Tu, Z. Shi, and Y. Liu, "Zero-shot image classification based on a learnable deep metric," *Sensors*, vol. 21, no. 9, p. 3241, 2021.

[13] W. Wang, V. W. Zheng, H. Yu, and C. Miao, "A survey of zero-shot learning: Settings, methods, and applications," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 2, 2019.

[14] Y. Xian, C. H. Lampert, B. Schiele, and Z. Akata, "Zero-shot learning—a comprehensive evaluation of the good, the bad and the ugly," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 9, pp. 2251–2265, 2018.

[15] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, and A. e. a. Askell, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[16] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa, "Large language models are zero-shot reasoners," *Advances in neural information processing systems*, vol. 35, pp. 22 199–22 213, 2022.

[17] S. Chen, Z. Hong, W. Hou, G.-S. Xie, Y. Song, and et al, "Transzero++: Cross attribute-guided transformer for zero-shot learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 11, pp. 12 844–12 861, 2023.

[18] D. Kondratyuk, L. Yu, X. Gu, J. Lezama, J. Huang, and et al, "Videopoet: A large language model for zero-shot video generation," 2024.

[19] N. Hollmann, S. Müller, K. Eggensperger, and F. Hutter, "TabPFN: A transformer that solves small tabular classification problems in a second," in *NeurIPS 2022 First Table Representation Workshop*, 2022.

[20] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, and et al, "Attention is all you need," in *Advances in Neural Information Processing Systems*, vol. 30. Curran Associates, Inc., 2017.

[21] A. Webson and E. Pavlick, "Do prompt-based models really understand the meaning of their prompts?" in *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Seattle, United States: Association for Computational Linguistics, 2022, pp. 2300–2344.

[22] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "Caltech-ucsd birds-200-2011 (cub-200-2011)," California Institute of Technology, Tech. Rep. CNS-TR-2011-001, 2011.

[23] Y. Atzmon and G. Chechik, "Adaptive confidence smoothing for generalized zero-shot learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11 671–11 680.