Aprendizaje incremental eficiente mediante enrutamiento por prototipos y entropía

Daniel Vila-Cruz[1][0009-0001-2116-4085],
Laura Morán-Fernández[1][0000-0001-6703-1846] y
Verónica Bolón-Canedo[1][000-0002-0524-6427]

Universidade da Coruña, CITIC, A Coruña, España d.vila1@udc.es

Resumen El olvido catastrófico es uno de los obstáculos fundamentales en el aprendizaje incremental: cuando una red neuronal se entrena sobre una nueva tarea, tiende a sobrescribir el conocimiento adquirido en tareas anteriores. En este trabajo presentamos un sistema que aborda este problema mediante una arquitectura modular de tres componentes principales: (1) un extractor de características congelado y desacoplado, (2) un conjunto de clasificadores entrenados de forma independiente empleando una función de pérdida que busca maximizar la entropía de muestras OOD, y (3) un selector de clasificadores, que determina qué clasificador emplear en inferencia, basándose en prototipos y entropía. Los experimentos realizados en los conjuntos de datos ImageNet-100, CIFAR-100 y CORe50 demuestran que nuestro modelo obtiene mejor exactitud que otros métodos del estado del arte, al mismo tiempo que reduce significativamente el tiempo de entrenamiento, convirtiéndolo en una solución práctica para aplicaciones del mundo real y en dispositivos integrados o de bajos recursos, donde la energía, la eficiencia y las rápidas adaptaciones al entorno son críticas.

Palabras clave: Aprendizaje incremental · Aprendizaje eficiente · Redes convolucionales · Olvido catastrófico · Algoritmos verdes.

1. Introducción

El aprendizaje incremental busca que un modelo pueda incorporar nuevo conocimiento sin degradar lo previamente aprendido. Existen numerosas motivaciones para emplear este paradigma. En primer lugar, la volatilidad de los datos. No siempre es posible conservar los datos originales, y en el futuro podrían perderse o modificarse. En segundo lugar, el alto consumo del entrenamiento. La alternativa clásica sería volver a entrenar un modelo desde cero cada vez que se quiera añadir nuevo conocimiento, lo que implica costes elevados en tiempo, energía y dinero. En tercer lugar, la necesidad de adaptar un sistema a nuevo conocimiento en un tiempo limitado, ya que no es viable detener el sistema durante largos períodos. En cuarto lugar, limitaciones relacionadas con la privacidad. Algunas regulaciones exigen que, si un usuario solicita borrar sus datos,

estos ya no sean accesibles. Por último, limitaciones físicas de almacenamiento, como ocurre en dispositivos de *Edge Computing*, donde los recursos son limitados y no es posible guardar grandes volúmenes de datos.

La principal limitación de esta área de investigación es el olvido catastrófico [10]. Dada la naturaleza de la mayoría de sistemas de inteligencia artificial, y más concretamente de aprendizaje profundo, el realizar entrenamientos con nuevos datos sobreescribe los anteriores, ya que el conocimiento se almacena en los pesos de los modelos. Por ello, al incorporar nuevo conocimiento, el rendimiento sobre tareas previamente aprendidas tiende a degradarse.

En este trabajo proponemos un sistema que emplea un extractor de características congelado, encargado de extraer representaciones que se mantienen fijas durante todo el entrenamiento. Sobre estas representaciones, se genera un nuevo clasificador ligero, especializado en cada nuevo conjunto de datos. Para poder decidir qué clasificador emplear en tiempo de inferencia, se almacenan prototipos, que buscan ser una representación de cada clase, y se selecciona el clasificador basándose en un enrutador por entropía. A diferencia de gran parte de la literatura reciente, nuestra propuesta prioriza simultáneamente la exactitud y la eficiencia computacional. La estrategia propuesta reduce significativamente el coste de adaptación a nuevas clases, evita la necesidad de volver a entrenar representaciones profundas y minimiza la cantidad de información almacenada de tareas anteriores, ya que únicamente se conservan prototipos en el espacio de características. Todas estas ventajas están alineadas con la perspectiva de la inteligencia artificial verde y tienen gran relevancia en el aprendizaje incremental, pues facilitan el despliegue eficiente de modelos en dispositivos con recursos limitados y permiten que el ciclo de vida de grandes modelos sea asumible tanto energética como temporal y medioambientalmente. Las principales contribuciones de este trabajo son:

1. Un sistema de aprendizaje incremental basado en un extractor de características congelado y clasificadores independientes, cuyo coste de entrenamiento por tarea no depende del número de tareas previamente aprendidas.
2. Una estrategia de selección de clasificadores basada en prototipos y entropía que permite identificar dinámicamente el clasificador más adecuado durante la inferencia.
3. Una función de pérdida basada en LogitNorm para favorecer predicciones con alta entropía, evitando un exceso de confianza del clasificador ante muestras diferentes a las vistas durante el entrenamiento y mejorando el mecanismo de enrutamiento.

2. Estado del arte

La literatura distingue habitualmente tres escenarios de aprendizaje incremental [11]. En el aprendizaje incremental por tareas (*task-incremental learning*, TIL), la identidad de la tarea está disponible en inferencia, lo que lo convierte en el escenario más sencillo. En el aprendizaje incremental por dominio (*domain-incremental learning*, DIL), el conjunto de clases permanece constante pero la

distribución de entrada varía entre tareas. El clasificador no necesita distinguir entre tareas, sino adaptarse al cambio del dominio. El escenario más exigente es el aprendizaje incremental por clases (*class-incremental learning*, CIL), donde cada nueva tarea añade clases totalmente nuevas y el modelo debe reconocer todas las clases vistas hasta el momento sin acceso a la identidad de la tarea durante la inferencia, lo que obliga a resolver simultáneamente el problema de identificar a qué tarea pertenece la muestra y el de clasificación. Este trabajo se centra exclusivamente en CIL.

Una distinción adicional relevante es la configuración de inicialización. Varios métodos adoptan una primera tarea de mayor tamaño ($B > 0$) para adaptar la representación del extractor de características antes de las actualizaciones incrementales, obteniendo una ventaja implícita que puede no ser viable cuando la distribución de clases no se conoce de antemano. En este trabajo adoptamos la configuración $B = 0$, donde todas las tareas incrementales contienen el mismo número de clases, garantizando una dificultad uniforme a lo largo de la secuencia y eliminando dicho privilegio inicial.

2.1. Estrategias frente al olvido catastrófico

Las estrategias propuestas en la literatura para mitigar el problema del olvido catastrófico pueden agruparse en varias familias. Los métodos basados en *replay* reutilizan muestras de tareas anteriores durante el entrenamiento de nuevas tareas. Aproximaciones como DER [14] o RMM [7] emplean el reentrenamiento para preservar representaciones previas. Estas estrategias requieren almacenar datos de tareas pasadas y reprocesarlos continuamente, lo que incrementa tanto el consumo de memoria como el coste computacional del entrenamiento. Otra línea relevante corresponde a las redes dinámicas, donde la arquitectura se expande progresivamente para acomodar nuevas tareas o clases. Métodos como DER [14], FOSTER [12] o MEMO [15] incrementan la capacidad del modelo mediante nuevos módulos o representaciones adicionales, reduciendo así la interferencia entre conocimiento antiguo y nuevo. El crecimiento progresivo de la arquitectura provoca mayores tiempos de entrenamiento, incremento del consumo energético y mayor complejidad en inferencia. Frente a los métodos con *replay*, recientemente ha surgido un creciente interés por los enfoques *exemplar-free*, especialmente relevantes en escenarios con restricciones de privacidad, memoria o almacenamiento. Entre ellos destacan SSRE [16] y FeTrIL [9], que buscan preservar la separabilidad de las representaciones sin necesidad de almacenar muestras antiguas.

Además de la exactitud, la eficiencia computacional se ha convertido en un aspecto cada vez más relevante en aprendizaje incremental. Como señalan Harun et al. [4], muchas aproximaciones requieren más cómputo que el equivalente entrenamiento tradicional, lo que los hace poco prácticos para numerosas aplicaciones en entornos reales. En ese estudio destaca REMID [5], obteniendo la mejor puntuación de eficiencia, pero su exactitud es peor que otros métodos como DER. El problema del balance entre coste computacional y exactitud del sistema es especialmente crítico en dispositivos de *Edge Computing*, robótica o

sistemas embebidos, donde existen fuertes limitaciones de energía, memoria y capacidad de cómputo. En este contexto, propuestas como CIFNet [3] buscan reducir el coste de adaptación mediante el uso de extractores de características preentrenados y congelados.

Otra cuestión importante en CIL es la configuración inicial del aprendizaje. Algunos métodos como FeTrIL o SSRE utilizan una tarea inicial significativamente mayor (configuración $B > 0$) para adaptar previamente el extractor de características antes de comenzar las actualizaciones incrementales. Aunque esta estrategia mejora el rendimiento, introduce una ventaja implícita poco realista en numerosos escenarios reales, donde la distribución completa de clases no está disponible de antemano. Por ello, configuraciones más estrictas como B=0, donde todas las tareas contienen el mismo número de clases, representan un escenario más exigente y realista para evaluar la capacidad de adaptación incremental de los modelos.

2.2. Posicionamiento del trabajo

Nuestra propuesta combina ideas procedentes de las redes dinámicas y de los enfoques basados en *replay*, pero buscando un equilibrio entre exactitud y eficiencia, buscando solventar las principales desventajas de dichos grupos. Primero, el extractor de características se congela completamente desde el inicio, de modo que la expansión se limita a clasificadores ligeros, con un coste de entrenamiento por tarea independiente del número de tareas ya aprendidas. Segundo, en inferencia se intenta emplear un único clasificador. Sin embargo, cuando la muestra cae en una región ambigua del espacio de características, se utiliza como máximo un número limitado de ellos. Tercero, el *replay* empleado es mínimo, pues únicamente se repiten prototipos de las características de las clases, no imágenes completas como en otros métodos.

3. Metodología propuesta

Proponemos un sistema para abordar el problema del olvido catastrófico en aprendizaje incremental, concretamente en CIL, priorizando la rapidez del modelo para adaptarse a nuevas clases. Los procesos de entrenamiento e inferencia están ilustrados en las Figuras 1 y 2. La arquitectura de nuestro sistema se organiza en tres componentes principales:

1. Un extractor de características preentrenado y congelado, que proyecta cada imagen a un espacio de representación fijo, y no participa durante el entrenamiento.
2. Un conjunto de clasificadores, donde cada tarea dispone de su propio clasificador lineal entrenado de forma independiente empleando una pérdida que maximiza la entropía de muestras OOD (*Out-of-Distribution*), mientras que mantiene la exactitud en las muestras ID (*In-Distribution*).
3. Un selector de clasificadores, que en inferencia determina qué clasificador o clasificadores deben participar en la clasificación de una muestra dada, basándose en prototipos y entropía.

3.1. Extractor de características

Se emplea una ResNet-34 preentrenada en ImageNet-1k [2] como extractor de características. Sus pesos permanecen congelados durante todo el proceso de aprendizaje incremental. Esto permite un espacio de características estable entre tareas, por tanto los prototipos generados en tareas anteriores siguen siendo válidos sin necesidad de adaptarlos o volverlos a calcular para cada tarea. Dado que el extractor no se actualiza, es posible separar la extracción de características del entrenamiento del clasificador, eliminando operaciones redundantes y reduciendo significativamente el coste computacional por tarea.

3.2. Clasificadores durante entrenamiento

Para cada tarea se instancia un clasificador lineal independiente. Esta aproximación evita modificar los pesos de clasificadores previamente aprendidos al incorporar nuevas tareas, mitigando así el olvido catastrófico. Sin embargo, al tratarse de clasificadores independientes, una muestra perteneciente a una tarea no vista puede producir predicciones sobreconfiadas. Para mitigar este comportamiento se emplea la pérdida LogitNorm [13], buscando mitigar la sobreconfianza del modelo ante clases no vistas:

$$\mathcal{R}_{\mathcal{L}}(f) = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{P}_{xy}} \left[\mathcal{L}_{\text{CE}} \left(\hat{f}(\mathbf{x};\theta), y \right) \right] \tag{1}$$

donde los logits normalizados vienen dados por:

$$\hat{f}(\mathbf{x};\theta) = \frac{f(\mathbf{x};\theta)}{\tau \|f(\mathbf{x};\theta)\|_2} \tag{2}$$

Nuestra propuesta incorpora una regularización sobre muestras fuera de distribución (OOD):

$$\mathcal{R}_{\mathcal{L}\text{-OOD}}(f) = \mathcal{R}_{\mathcal{L}}(f) + \lambda \mathcal{R}_{\text{OOD}}(f) \tag{3}$$

con:

$$\mathcal{R}_{\text{OOD}}(f) = \mathbb{E}_{\tilde{\mathbf{x}}\sim\mathcal{P}_{\text{OOD}}} \left[D_{\text{KL}} \left(U \parallel p_\theta(y|\tilde{\mathbf{x}}) \right) \right] \tag{4}$$

donde D_{KL} representa la divergencia Kullback-Leibler, $p_\theta(y|\tilde{\mathbf{x}})$ corresponde a la distribución de probabilidad obtenida tras aplicar la función $softmax$ sobre los logits normalizados, y U representa la distribución uniforme sobre las clases. Minimizar esta divergencia favorece que las predicciones sobre muestras OOD se aproximen a una distribución uniforme, promoviendo así salidas de alta entropía y reduciendo la sobreconfianza ante datos no vistos. Esta propiedad se emplea posteriormente en el mecanismo de selección de clasificadores descrito en la Sección 3.3.

En los experimentos se ha empleado el valor de temperatura $\tau = 0{,}04$ propuesto en el trabajo original [13]. En cuanto a las muestras OOD, estas se generan sintéticamente en el espacio de características combinando dos estrategias:

(i) ruido global respecto a la distribución de características de la tarea, generado a partir de direcciones aleatorias escaladas según la desviación estándar respecto al centroide global, y (ii) perturbaciones locales sobre prototipos de tareas anteriores mediante la adición de ruido gaussiano. Para la primera tarea, dado que no se dispone de prototipos previos, únicamente se emplea la primera estrategia.

3.3. Enrutamiento en inferencia

Para determinar qué clasificador posee el conocimiento más adecuado para asignar una nueva muestra a una clase concreta, se compara su representación con prototipos generados durante el entrenamiento. Estos prototipos se obtienen mediante K-Means sobre las características de cada clase, de forma que cada prototipo corresponde a un centroide representativo de dicha clase.

El número de prototipos generados dependerá de la configuración del experimento. Para poder realizar una comparativa justa con otros métodos, se ha buscado generar 2000 prototipos en total, de forma que si un experimento consta de 100 clases, divididas en 20 tareas con 5 clases cada una, se generan 20 prototipos por clase.

La similitud coseno entre características y prototipos es efectiva cuando las tareas están bien separadas en el espacio de representación, pero puede degradarse cuando diferentes tareas ocupan regiones solapadas. Para abordar este problema, se calcula una matriz de exactitud global mediante el estimador de Ledoit-Wolf sobre los prototipos de cada tarea. La puntuación de afinidad de una muestra con una tarea se calcula como la log-suma-exponencial ($LogSumExp$) sobre los prototipos. Esta formulación suele ser más discriminativa con respecto a la similitud coseno cuando el solapamiento entre tareas es significativo. Estas puntuaciones se normalizan en probabilidades. Si la diferencia de probabilidad entre la tarea más probable y la segunda es mayor a un umbral δ, se emplea únicamente el clasificador con mayor probabilidad. En caso contrario, se emplean los C clasificadores con mayor puntuación y el que menor entropía tenga será seleccionado. El umbral δ dependerá de la similitud de las clases entre tareas, de forma empírica hemos decidido emplear $\delta = 0{,}1$. En cuanto al número de clasificadores seleccionados cuando hay incertidumbre, se ha optado por emplear $C = 2$ cuando hay 5 muestras por tarea, y $C = 5$ cuando hay más. En caso de emplear C clasificadores, se seleccionará como válido el que menor entropía tenga en el *softmax* de la clasificación. Esto es posible gracias a haber empleado una función de pérdida que intente maximizar la entropía cuando los clasificadores se encuentren con una muestra OOD.

4. Configuración de los experimentos

Los experimentos se realizaron sobre tres conjuntos de datos de referencia en CIL: ImageNet-100 (subconjunto de 100 clases de ImageNet [2]), CIFAR-100 [6] y CORe50 [8], del cual, al tratarse de secuencias de vídeo, se ha extraído

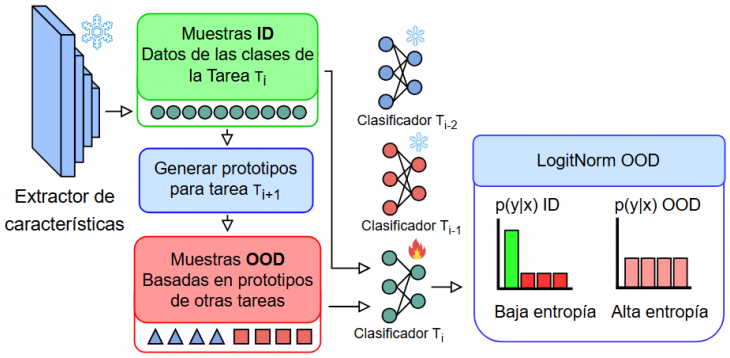


Figura 1. Representación del proceso de entrenamiento del sistema propuesto.

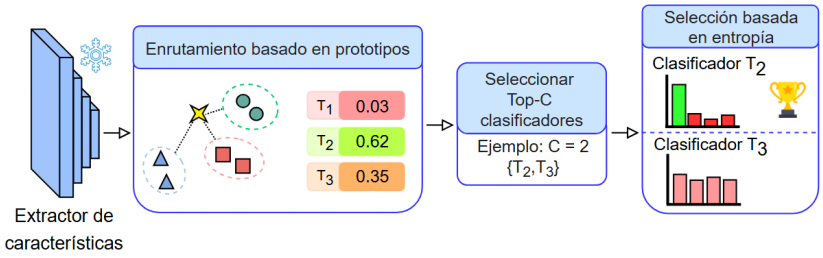


Figura 2. Representación del proceso de inferencia del sistema propuesto.

una imagen cada 10 fotogramas. La Tabla 1 resume sus estadísticas. Para cada conjunto de datos se evaluaron dos configuraciones de división de los datos incrementalmente: la configuración A con más tareas y menos clases por tarea, y la configuración B con menos tareas y más clases por tarea. Las clases se asignaron aleatoriamente a las tareas para garantizar diversidad, de modo que clases similares pueden aparecer en tareas distintas, aumentando la dificultad del enrutamiento.

Se ha comparado el método propuesto con *fine-tuning* secuencial sin mitigación del olvido, RMM FOSTER [7,12], DER [14] y CIFNet [3]. Todos los métodos que emplean un extractor de características usan el mismo, ResNet34 preentrenado en ImageNet-1k. Si bien esto puede ofrecer una ventaja a los métodos que usan este extractor en los experimentos con ImageNet-100, ya que tiene conocimiento concreto de los datos, hemos considerado oportuno añadir este conjunto de datos por dos motivos: (1) evaluar nuestro modelo con volúmenes más grandes de datos, y (2) comparar cómo se comportan diferentes arquitecturas cuando el extractor ya tiene conocimiento sobre el dominio. Las métricas reportadas son la exactitud media final sobre datos de todas las tareas, el tiempo de entrenamiento total, el consumo energético (kWh) y las emisiones equivalentes de CO_2

(kg CO_2-eq), medidas con CodeCarbon [1]. Los experimentos se ejecutaron en una GPU NVIDIA GeForce RTX 3090 Ti con CPU Intel Core i7-12700K.

Tabla 1. Especificaciones y configuraciones de los conjuntos de datos utilizados en los experimentos.

	ImageNet100		**CIFAR100**		**CORe50**	
#Muestras	135 000		60 000		16 500	
#Clases	100		100		50	
Configuración	**A**	**B**	**A**	**B**	**A**	**B**
#Tareas	20	5	20	5	10	5
#Clases/Tarea	10	10	5	20	5	10

5. Resultados experimentales

Exactitud. La Tabla 2 muestra que el método propuesto obtiene la mayor exactitud en todos los escenarios evaluados, con una diferencia de 6.2 pp (puntos porcentuales) sobre CIFNet, el segundo mejor método. Tal y como cabría esperar, nuestro modelo destaca en ImageNet-100, al disponer de un extractor de características entrenado en el mismo dominio. Sin embargo, cabe destacar que, aunque CIFNet también cuenta con esta ventaja, sus resultados son, promediando ambas configuraciones, 7.53 pp por debajo de los nuestros. En CIFAR-100, donde las clases se pueden agrupar en 10 grandes familias, y por tanto existe un gran solapamiento de características entre diferentes tareas, se puede apreciar una disminución de la exactitud. Esta limitación no solo afecta a nuestro método, aún así, la distancia con CIFNet se mantiene en 7.8 pp. Finalmente, el rendimiento del método es prácticamente invariante entre configuraciones A y B (diferencia < 0.2 pp en todos los conjuntos de datos), lo que evidencia robustez frente a distintas granularidades de partición incremental.

Eficiencia computacional y huella energética. Las Tablas 3 y 4 muestran la ventaja más notable del método propuesto: el tiempo de entrenamiento se reduce en uno o dos órdenes de magnitud respecto al resto de modelos, siendo incluso más rápido que el propio *fine-tuning* secuencial. En ImageNet-100 con configuración A, nuestro método completa el entrenamiento en 7.66 minutos frente a los 1354 de DER y los 71.5 de CIFNet. Esta reducción se traslada directamente al consumo energético y a las emisiones de CO_2: nuestro método consume de media 0.009 kWh frente a los 4.14 kWh de DER, una reducción de $\times 460$. Estos resultados son consecuencia directa del desacople del extractor de características: al congelar el extractor de características y entrenar únicamente clasificadores lineales ligeros, el coste por tarea es despreciable y no crece con el número de tareas ya aprendidas, a diferencia de los métodos.

Tabla 2. Comparación de exactitud (%) entre métodos en los distintos conjuntos de datos. Mejores resultados resaltados en negrita.

Método	Config.	ImageNet100	CIFAR100	CORe50	Promedio
Fine-tuning	A	4.64	5.25	19.11	9.67
RMM-FOSTER	A	59.46	51.10	38.82	49.79
DER	A	63.66	57.34	51.33	57.44
CIFNet	A	78.10	59.26	72.22	69.86
Nuestro método	A	**85.46**	**67.46**	**75.53**	**76.15**
Fine-tuning	B	9.34	17.14	19.11	15.20
RMM-FOSTER	B	66.10	63.01	48.76	59.29
DER	B	64.40	62.67	47.61	58.23
CIFNet	B	77.92	59.98	72.33	70.08
Nuestro método	B	**85.62**	**67.36**	**75.84**	**76.27**

Tabla 3. Comparación de tiempos de entrenamiento (minutos) entre métodos en los distintos conjuntos de datos. Mejores resultados resaltados en negrita.

Método	Config.	ImageNet100	CIFAR100	CORe50	Promedio
Fine-tune	A	188.93	14.17	35.64	79.58
RMM-FOSTER	A	935.67	76.36	401.45	471.16
DER	A	1354.40	102.44	295.55	584.13
CIFNet	A	71.50	15.13	3.63	30.09
Nuestro método	A	**7.66**	**1.57**	**0.54**	**3.26**
Fine-tune	B	184.20	10.91	31.61	75.57
RMM-FOSTER	B	757.55	37.55	255.92	350.34
DER	B	742.22	28.50	134.76	301.83
CIFNet	B	70.02	13.90	3.21	29.04
Nuestro método	B	**6.83**	**1.37**	**0.50**	**2.90**

Análisis de la selección de clasificadores. La Tabla 5 descompone el error del sistema en sus dos fuentes: fallos de enrutamiento y fallos de clasificación cuando el enrutamiento es correcto. La exactitud de clasificación condicionada a un enrutamiento correcto (Clf|Rout) es consistentemente alta en todos los escenarios (84.5–97.6 %), lo que indica que los clasificadores por tarea son efectivos cuando reciben el ejemplo correspondiente. El cuello de botella del sistema es el selector: la brecha entre Rout@C (el porcentaje de acierto de que el clasificador correcto esté entre los C clasificadores seleccionados) y Clf E2E (la exactitud final del sistema en conjunto) muestra que reducir los errores de enrutamiento es la vía de mejora con mayor impacto potencial. Además, puede observarse que Clf E2E tiene valores muy aproximados al producto entre Rout@C y Clf|Rout, lo que sugiere que, cuando el clasificador correcto se encuentra entre los candidatos seleccionados, el mecanismo basado en entropía suele ser capaz de seleccionar al experto adecuado. Este comportamiento indica que la entropía actúa como un indicador fiable de confianza para discriminar entre clasificadores ID y OOD. En

Tabla 4. Comparación de energía consumida (kWh) y emisiones equivalentes de CO_2 (kg CO_2-eq) durante el entrenamiento. Los mejores resultados se resaltan en negrita.

		ImageNet100		**CIFAR100**		**CORe50**		**Promedio**	
Método	**Conf.**	**kWh**	**CO_2**	**kWh**	**CO_2**	**kWh**	**CO_2**	**kWh**	**CO_2**
Fine-tune	A	1.1362	0.1981	0.0681	0.0124	0.1862	0.0323	0.4635	0.0807
RMM-FOSTER	A	6.2553	1.0892	0.4299	0.0754	2.4026	0.4185	3.0291	0.5273
DER	A	9.9726	1.7365	0.6667	0.1164	1.7879	0.3111	4.1422	0.7210
CIFNet	A	0.2718	0.0479	0.0635	0.0112	0.0135	0.0024	0.1168	0.0200
Nuestro método	A	**0.0202**	**0.0035**	**0.0054**	**0.0008**	**0.0023**	**0.0004**	**0.0086**	**0.0014**
Fine-tune	B	1.1527	0.2011	0.0610	0.0113	0.1869	0.0323	0.4667	0.0813
RMM-FOSTER	B	5.1867	0.9037	0.2278	0.0415	1.5447	0.2692	2.3194	0.4043
DER	B	5.3286	0.9274	0.1724	0.0327	0.8042	0.1405	2.1019	0.3663
CIFNet	B	0.2646	0.0469	0.0581	0.0102	0.0028	0.0112	0.1080	0.0223
Nuestro método	B	**0.0191**	**0.0033**	**0.0054**	**0.0008**	**0.0021**	**0.0004**	**0.0089**	**0.0014**

la Figura 3 puede apreciarse una comparativa entre tres funciones de pérdida diferentes: entropía cruzada, LogitNorm y nuestra adaptación de LogitNorm. La figura muestra la confianza que tiene cada clasificador, una vez entrenado, con los datos de las diferentes tareas. Esta confianza se calcula como $1 -$ entropía, e idealmente ha de ser alta en la tarea donde el clasificador ha sido entrenado (diagonal) y baja en las demás, es decir, en los casos de OOD. Los resultados muestran como nuestra aproximación es la que mejor separa las muestras ID de las OOD, con una media de 22.9 puntos, frente a los 15.1 obtenidos con entropía cruzada y los 11.0 obtenidos con LogitNorm.

Tabla 5. Exactitud del mecanismo de enrutamiento desglosada en % de veces que el clasificador correcto se encuentra entre los C seleccionados (Rout@C), exactitud del clasificador correcto cuando es seleccionado (Clf|Rout) y exactitud global del sistema (Clf E2E).

Conjunto de datos	**Config.**	**Rout@C**	**Clf|Rout**	**Clf E2E**
ImageNet100	A	87.5	97.6	85.5
CIFAR100	A	79.8	84.5	67.5
CORe50	A	82.8	91.2	75.5
ImageNet100	B	88.3	96.9	85.6
CIFAR100	B	75.8	88.9	67.4
CORe50	B	84.2	90.1	75.8

6. Conclusiones

Se ha presentado un sistema de aprendizaje incremental de clases que combina un extractor de características congelado, clasificadores lineales por tarea

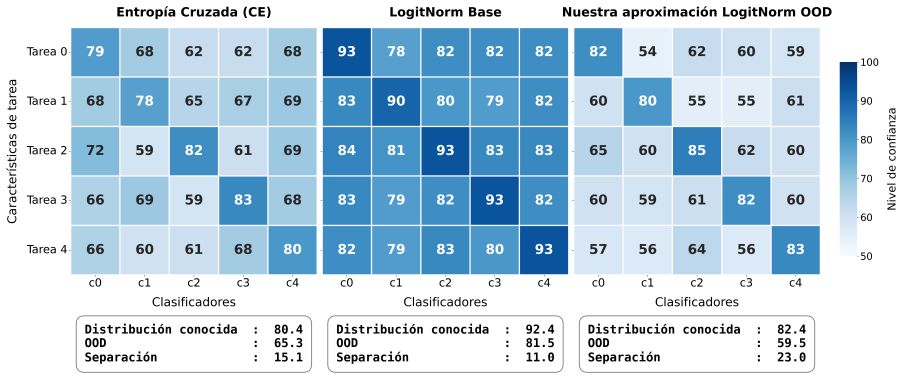


Figura 3. Ejemplo de confianza de los clasificadores en las diferentes tareas de CIFAR-100 configuración B, entrenados con entropía cruzada, LogitNorm, y nuestra adaptación de LogitNorm. La diagonal indica la tarea donde ha sido entrenado cada clasificador.

entrenados con una propuesta de pérdida LogitNorm que busca maximizar la entropía de muestras OOD, y un selector de clasificadores que emplea prototipos y entropía. El sistema mitiga el olvido catastrófico al aislar el entrenamiento de cada clasificador, y consta de un *replay* ligero, pues solo se emplean prototipos de características y con un coste de entrenamiento por tarea independiente del número de tareas ya aprendidas. Los experimentos sobre ImageNet-100, CIFAR-100 y CORe50 demuestran que el método supera a otros métodos del estado del arte en exactitud, reduce el tiempo de entrenamiento y el consumo energético en uno o dos órdenes de magnitud, lo que lo hace especialmente adecuado para entornos con recursos limitados.

Como trabajo futuro, se plantea mejorar la selección de clasificadores inicial, pues es el punto limitante de la exactitud del sistema. Para ello, se propone experimentar con prototipos más discriminativos, ya sea mediante una proyección de las características o un mecanismo de selección de características.

Agradecimientos Esta investigación ha sido financiada por el Ministerio de Ciencia e Innovación de España (Subvención PID2023-147404OB-I00 / AEI / 10.13039 / 501100011033), junto con "NextGenerationEU"/PRTR del Ministerio de Transformación Digital y Función Pública bajo la subvención TSI-100925-2023-1. CITIC, como miembro de la Red CIGUS, recibe subvenciones de la Xunta de Galicia y del Programa Operativo FEDER Galicia 2021-2027 (Subvenciones ED431G 2023/01 y ED431C 2026/38).

Referencias

1. Courty, B., Schmidt, V., Luccioni, S., Goyal, K., Coutarel, M., Feld, B., Lecourt, J., Connell, L., Saboni, A., Inimaz, supatomic, Léval, M., Blanche,

L., Cruveiller, A., ouminasara, Zhao, F., Joshi, A., Bogroff, A., de Lavoreille, H., Laskaris, N., Abati, E., Blank, D., Wang, Z., Catovic, A., Alençon, M., Stęchły, M., Bauer, C., de Araújo, L.O.N., JPW, MinervaBooks: mlco2/codecarbon: v2.4.1 (May 2024). https://doi.org/10.5281/zenodo.11171501, https://doi.org/10.5281/zenodo.11171501
2. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. IEEE (2009)
3. Dopico-Castro, A., Fontenla-Romero, O., Guijarro-Berdiñas, B., Alonso-Betanzos, A.: Efficient single-step framework for incremental class learning in neural networks. arXiv preprint arXiv:2509.11285 (2025)
4. Harun, M.Y., Gallardo, J., Hayes, T.L., Kanan, C.: How efficient are today's continual learning algorithms? In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2431–2436 (2023)
5. Hayes, T.L., Kafle, K., Shrestha, R., Acharya, M., Kanan, C.: Remind your neural network to prevent catastrophic forgetting. In: European conference on computer vision. pp. 466–483. Springer (2020)
6. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
7. Liu, Y., Schiele, B., Sun, Q.: Rmm: Reinforced memory management for class-incremental learning. Advances in neural information processing systems **34**, 3478–3490 (2021)
8. Lomonaco, V., Maltoni, D.: Core50: a new dataset and benchmark for continuous object recognition. In: Conference on robot learning. pp. 17–26. PMLR (2017)
9. Petit, G., Popescu, A., Schindler, H., Picard, D., Delezoide, B.: Fetril: Feature translation for exemplar-free class-incremental learning. In: Proceedings of the IEEE/CVF winter conference on applications of computer vision. pp. 3911–3920 (2023)
10. Van de Ven, G.M., Soures, N., Kudithipudi, D.: Continual learning and catastrophic forgetting, p. 153–168. Elsevier (2025). https://doi.org/10.1016/b978-0-443-15754-7.00073-0
11. Van de Ven, G.M., Tolias, A.S.: Three scenarios for continual learning. arXiv preprint arXiv:1904.07734 (2019)
12. Wang, F.Y., Zhou, D.W., Ye, H.J., Zhan, D.C.: Foster: Feature boosting and compression for class-incremental learning. In: European conference on computer vision. pp. 398–414. Springer (2022)
13. Wei, H., Xie, R., Cheng, H., Feng, L., An, B., Li, Y.: Mitigating neural network overconfidence with logit normalization. In: International conference on machine learning. pp. 23631–23644. PMLR (2022)
14. Yan, S., Xie, J., He, X.: Der: Dynamically expandable representation for class incremental learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 3014–3023 (2021)
15. Zhou, D.W., Wang, Q.W., Ye, H.J., Zhan, D.C.: A model or 603 exemplars: Towards memory-efficient class-incremental learning. arXiv preprint arXiv:2205.13218 (2022)
16. Zhu, K., Zhai, W., Cao, Y., Luo, J., Zha, Z.J.: Self-sustaining representation expansion for non-exemplar class-incremental learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 9296–9305 (2022)