ISSN 1870-4069

# Análisis de impacto en clasificadores CNNs ante la evaluacion de imagenes con perturbaciones naturales

Robin A. Rojas-Alvarez[1], Ivan Reyes-Amezcua[2],
Andres Mendez-Vazquez[2]

[1] Universidad de Guadalajara,
Ingeniería en Nanotecnología,
Mexico

[2] Centro de Investigación y de Estudios Avanzados,
Departamento de Computación,
México

{ivan.reyes, andres.mendez}@cinvestav.mx,
robin.rojas@alumnos.udg.mx

**Resumen.** El desarrollo de algoritmos capaces de robustecer las redes neuronales profundas se ha convertido en parte esencial de la metodología en la creación de soluciones actuales. Sin embargo, estando en la era del crecimiento y comprensión de las capacidades de las IA, se ha promovido la funcionalidad sobre la seguridad. En este trabajo, se expone el uso de redes neuronales preentrenadas para observar su robustez con el algoritmo de entrenamiento de conjuntos de datos basados en CIFAR-C, identificando puntos clave en la arquitectura para mejorar la respuesta y el comportamiento de las DNNs ante perturbaciones naturales.

**Palabras clave:** Robustecimiento de algoritmos, CIFAR-C, Fine-tunning, CNN, DNN, preentrenamiento, corrupciones naturales.

## Impact Analysis on CNN Classifiers in the Evaluation of Naturally Disturbed Images

**Abstract.** The development of algorithms capable for robust deep neural networks has become an essential part of the methodology in the creation of current solutions. However, being in the era of growth and understanding the AI capabilities, functionality has been promoted over security. In this paper, we expose the use of pre-trained neural networks to observe their robustness with the CIFAR-C based on dataset training algorithm, identifying key points in the architecture to improve the response and behavior of DNNs to natural disturbances.

**Keywords:** Algorithm robustness, CIFAR-C, Fine-tunning, CNN, DNN, preentrain, natural corruptions.

# 1. Introducción

El crecimiento exponencial de las redes neuronales profundas (DNNs, por sus siglas en inglés) ha generado grandes soluciones, destacando en el rubro de las tareas hechas por visión artificial, a su vez se han encontrado nuevas formas de perturbar los algoritmos con alteraciones naturales. En análisis iniciales de casos en DNNs [13] han demostrado que perturbaciones, muchas veces imperceptibles ante el ojo humano, de las imágenes provocaban un cambio en la predicción de los modelos, que pasaba de ser correcta a incorrecta. Los trabajos posteriores de han demostrado la susceptibilidad de las DNNs a las corrupciones naturales no adversarias.

Se han observado diferencias significativas en el rendimiento de las DNNs entre las evaluaciones en condiciones de imagen limpias y las degradadas por perturbaciones naturales (a menudo hasta un 30-40 % de disminución de la precisión) [2], siendo estos resultados motivo de preocupación sobre la fiabilidad de las DNNs a medida que se integran en sistemas con riesgos sociales y de seguridad cada vez mayores. La gran vulnerabilidad de los modelos a los daños infrecuentes y naturales de las imágenes sugiere la necesidad de volver a dar prioridad a nuestra comprensión del rendimiento de los modelos con datos y perturbaciones naturales antes de centrarnos en la resistencia a escenarios de ataques de adversarios, robusteciendo así los algoritmos a estas perturbaciones iniciales.

# 2. Antecedentes y definiciones

## 2.1. Redes neuronales profundas y robustecimiento

Las redes neuronales profundas son una composición de funciones y capas computacionales que mapean desde el espacio de entrada en el dominio de la imagen a una predicción. Estas redes están muy parametrizadas, por lo que requieren grandes conjuntos de datos y/o tratamiento de los mismos, en combinación con la optimización de los parámetros (a menudo descenso de gradiente estocástico)[2]. Este artículo se enfocara en las DNNs con aplicaciones en visión artificial.

La robustez en general es un término que se ah adoptado a una serie de interpretaciones en la en la comunidad de la visión por ordenador, incluyendo, entre otras, el rendimiento bruto de la tarea en conjuntos de pruebas, el mantenimiento del rendimiento de la tarea en entradas manipuladas/modificadas, la generalización entre dominios y la resistencia a ataques de adversarios. Sin embargo, todas estas pueden ser características deseadas de la robustez, ya que existen diferentes enfoques, dependiendo el grado de optimización y el tipo de ataque resistente, tomando esto en cuenta, se generalizará a través de la resistencia a alteraciones naturales de los conjuntos de datos que alimentan al algoritmo.

Pruebas recientes sugieren que la robustez de las DNNs se beneficia enormemente de conjuntos de datos a gran escala, ya que contienen más variaciones que podrían ocurrir en el mundo real, llenando la brecha de distribución entre los datos de entrenamiento y los de prueba. Sin embargo, el coste de obtener un conjunto de datos grande y bien definido puede ser excesivamente alto. Por ello, los investigadores generan sintéticamente datos con diferentes aumentos para aumentar la variedad de

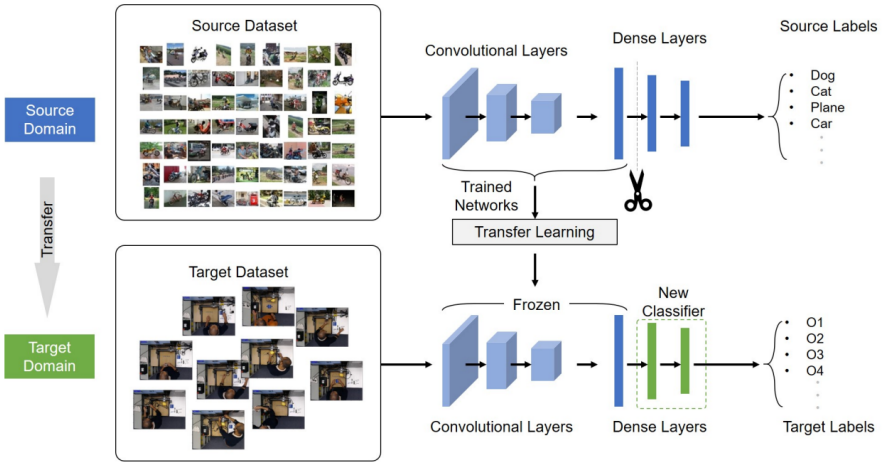



**Fig. 1.** Arquitectura del modelo de tranfer learning, imágen tomada de [14].

datos y disminuir la brecha de distribución entre los datos de entrenamiento y los de prueba. entre los datos de entrenamiento y los de prueba [16, 14] Mientras que las perturbaciones naturales abarcan desde simples transformaciones como voltear las imágenes, recortar, rotar, trasladar, alteración del color, mejora de bordes, ruido gaussiano hasta redes neuronales generativas adversarias (GANs)

## 2.2. Fine-tuning y transfer learning

El meta aprendizaje, o aprender a aprender, utiliza los conocimientos previos de una de tareas, algoritmos y evaluaciones de modelos con el fin de obtener mejores resultados, más rápidos y más eficientes cuando se aplican a datos que no se han visto antes. a datos que no se han visto antes [3]. Un problema común del meta aprendizaje es la recomendación de algoritmos, donde dado un conjunto de instancias de problemas $P$ de una distribución $D$, un conjunto $A$ de algoritmos y una medida de rendimiento $m$: $P$ x $A \rightarrow \mathbb{R}$, el problema de recomendación de algoritmos consiste en encontrar un mapeo $f : P \rightarrow A$ que optimiza la medida de rendimiento esperada $m$ para instancias $P$ con una distribución $D$ [11]. Se ofrece una definición formal del aprendizaje por transferencia en términos de dominios y tareas de aprendizaje.

Dado un dominio de origen $D_s$ un dominio de destino $D_t$, y unas tareas de aprendizaje $T_s$ y $T_t$, el aprendizaje por transferencia pretende mejorar el rendimiento de $T_t$ con conocimientos obtenidos de un dominio $D_s$ y una tarea de aprendizaje $T_s$ diferentes pero relacionados, donde $D_s \neq D_t$, o $T_s \neq T_t$. [9, 10]. En cambio que el ajuste fino de un modelo preentrenado es un enfoque común para llevar a cabo el aprendizaje profundo a medida que los modelos de base están disponibles. Si bien este enfoque mejora el aprendizaje supervisado en varios casos, también se ha observado un exceso de ajuste durante el ajuste fino supervisado en varios casos, también se ha observado sobreajuste durante el ajuste fino, Figura 1 .

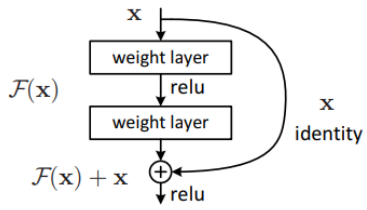



**Fig. 2.** Bloque de construcción para arquitectura ResNet, tomado de [4].

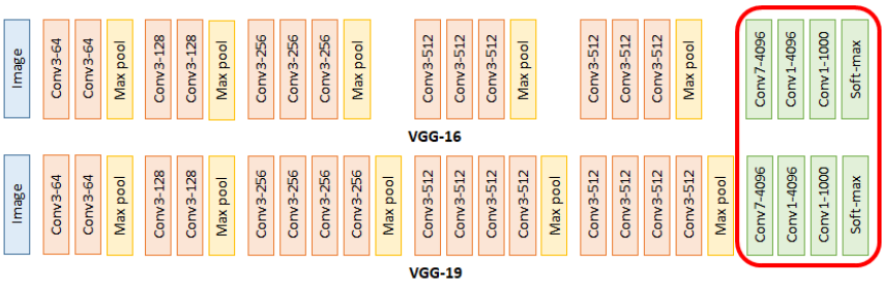



**Fig. 3.** Arquitectura de una CNN VGG16 y VGG19, tomado de [12].

Entender la causa del sobreajuste es un reto, ya que diseccionar el problema en la práctica requiere una medición precisa de los errores de generalización de las redes neuronales profundas. de las redes neuronales profundas [6].

## 3. Modelos de redes neuronales convolucionales

Los modelos de redes neuronales convolucionales, CNN por sus siglas en inglés, son arquitecturas muy eficientes, gracias a que cuentan con tres capas principales, las cuales son:

– Capa convolucional.

– Capa de agrupamiento.

– Capa “Fully-Connected”, FC.

La capa convolucional es la primera capa de una red convolucional. Mientras que las capas convolucionales pueden ir seguidas de capas convolucionales adicionales o capas de agrupamiento, la capa totalmente conectada es la última capa. Con cada capa, la CNN aumenta su complejidad, identificando mayores porciones de la imagen. Las primeras capas se centran en características simples, como colores y bordes. A medida que los datos de la imagen avanzan por las capas de la CNN, ésta empieza a reconocer elementos o formas más grandes del objeto hasta que finalmente identifica el objeto deseado, mejorando así su desempeño en la búsqueda y reconocimiento de entradas de audio, diálogo (“speach”), e incluso en imágenes, siendo esta una gran ventaja y área de interés cuando se trata del robustecimiento de algoritmos contra ataques.

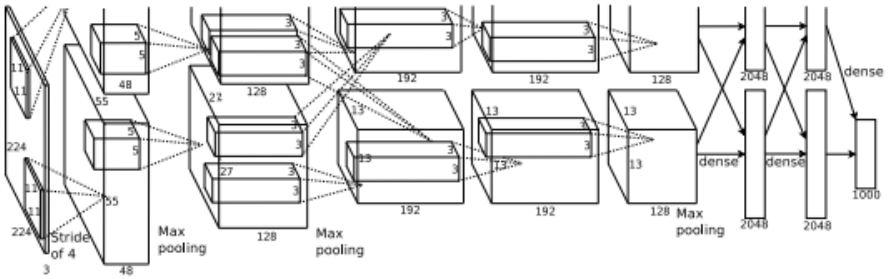



**Fig. 4.** Ilustración de arquitectura de una CNN, donde muestra explícitamente la delimitación de responsabilidades entre las dos GPU. Una GPU ejecuta las capas de la parte superior de la figura, mientras que la otra ejecuta las capas de la parte inferior. Las GPU sólo se comunican en determinadas capas. tomada de [8].

### 3.1. ResNet

Residual Network, también conocida como ResNet, es una CNN que acepta una resolución de 224x224, En esta infraestructura en lugar de esperar que cada una de las capas apiladas se ajuste directamente a una cartografía subyacente deseada, se deja explícitamente que estas capas se ajusten a una cartografía residual, Figura 2.

Formalmente, denotando la cartografía subyacente deseada como $\mathcal{H}(x)$, dejando que las capas no lineales apiladas se ajusten a otra cartografía de $\mathcal{F}(x) := \mathcal{H}(x) - x$. El mapeo original se refunde en $\mathcal{F}(x) + x$. Esta infraestructura democratizó el concepto de "residual learning" y "skip connections" mejorando en gran medida la posibilidad de entrenar de manera más profunda los modelos futuros, encontrándose en diferentes presentaciones, siendo de las más conocida:

– ResNet 18.

– ResNet 50.

– ResNet 101.

[NOTA] Estas mismas han sido utilizadas para el análisis de robustez de la arquitectura en este estudio.

### 3.2. VGG

VGG, Visual Geometry Group, es un arquitectura CNN clásica, que consta de diferentes capas convolucionales y un numero variable, de acuerdo al modelo que mejor ajuste, de capas "fully-connected", siendo lo más común encontrar esta arquitectura con un total de 16 y 19 capas, siendo llamada VGG16 y VGG19 respectivamente. La red VGG se introdujo utilizando capas convolucionales apiladas unas sobre otras a profundidades crecientes. Mediante la agrupación máxima, se reduce el tamaño del volumen. Luego le siguen dos capas totalmente conectadas, cada una con 4.096 nodos y, a continuación, un clasificador softmax.

**Tabla 1.** Conjuntos de datos de referencia para cambios en la distribución de datos en corrupciones de imágenes en el dataset CIFAR.

| Categoría | Dataset | Tipos de variaciones en las imágenes |
|---|---|---|
| Corrupción | CIFAR-C | **Ruido** (Gaussiano, Impulso, Disparo, Moteado); |
| | | **Borroso** (Desenfoque, Cristal, Gaussiano, Movimiento, Zoom); |
| | | **Meteorológico** (Niebla, Escarcha, Nieve, Salpicaduras); |
| | | **Digital** (Brillo, Contraste, Elástico, Compresión Jpeg, Pixelado, Saturado) |

**Tabla 2.** Diferencias de precisión obtenidas en modelos Convolucionales a partir de un datatest basado en cifar10-C comparados con un datatest sin corrupciones.

| Diferencia de precisión | ResNet18 | ResNet50 | ResNet101 | VGG16 | VGG19 | AlexNet |
|---|---|---|---|---|---|---|
| Modelo sin preentrenamiento | 57 % | 58 % | 51 % | 35 % | 37 % | 3 % |
| Modelo con preentrenamiento | 85 % | 81 % | 86 % | 81 % | 75 % | 63 % |

VGG en todas sus capas utiliza filtros de convolución muy pequeños ($3 \times 3$) y el paso convolucional es igual a 1 píxel para reducir el número de parámetros en esta red profunda [12], esta arquitectura se puede apreciar, en el modelo VGG16 y VGG19, en la Figura 3.

### 3.3. AlexNet

AlexNet es una CNN que contiene ocho capas con pesos; las cinco primeras son convolucionales y las tres restantes están totalmente conectadas. La salida de la última capa totalmente conectada se alimenta a un softmax de 1000 vías que produce una distribución sobre las 1000 etiquetas de clase. La primera capa convolucional filtra la imagen de entrada de $224\times224\times3$ con 96 núcleos de tamaño $11\times11\times3$ con un intervalo de 4 píxeles (ésta es la distancia entre los centros del campo receptivo de las neuronas vecinas en un mapa de núcleos).

La segunda capa convolucional toma como entrada la salida (de respuesta normalizada y agrupada) de la primera capa convolucional y la filtra con 256 núcleos de tamaño $5\times5\times48$. La tercera, cuarta y quinta capas convolucionales están conectadas entre sí sin capas intermedias de agrupación o normalización. La tercera capa convolucional tiene 384 núcleos de tamaño $3\times3\times256$ conectados a las salidas (normalizadas, agrupadas) de la segunda capa convolucional. La cuarta capa convolucional tiene 384 núcleos de tamaño $3\times3\times192$ , y la quinta capa convolucional tiene 256 núcleos de tamaño $3\times3\times192$.

Las capas totalmente conectadas tienen 4096 neuronas cada una, Figura 4 [8]: Esta arquitectura permite en gran medida disminuir el “Overfitting”, un comportamiento indeseable del aprendizaje automático que se produce cuando el modelo de aprendizaje automático ofrece predicciones precisas para los datos de entrenamiento, pero no para los datos nuevos.

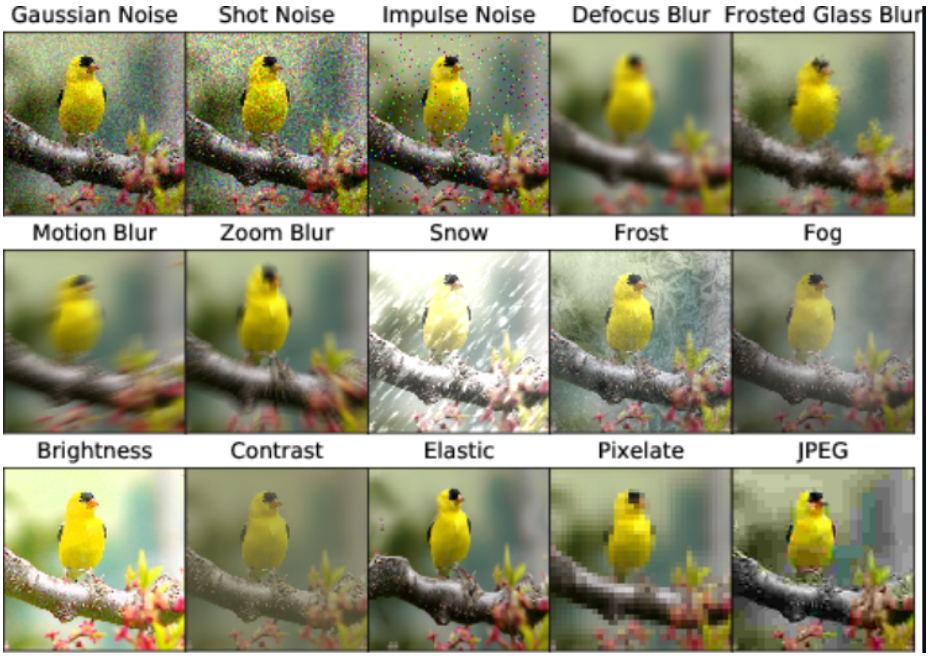



**Fig. 5.** Ejemplos de corrupciones aplicadas a imágenes, tomado de [5].

## 4. Conjuntos de datos corruptos

Cifar-C es una variante del conjunto de datos CIFAR, el cual contiene 19 tipos de corrupciones. Sin embargo, este tipo de corrupciones no son suficientes para representar todas las posibles variaciones que pueden ocurrir en el mundo real [15], tabla 1. Sin embargo, estas nos permiten vislumbrar cómo respondería una red neuronal convolucional ante ataques por envenenamiento, también llamados "Poisoning", observando así la precisión, pérdida y robustez de las arquitecturas ante entradas corruptas naturalmente.

## 5. Comportamiento de arquitecturas CNN ante un dataset corrupto con y sin preentrenamiento

### 5.1. Configuración de las variables de experimentación

Se siguió la configuración básica para la implementación en Pytorch de un dataset basado en CIFAR-10 dividiendo y descargando la información del datatrain, siguiendo la metodología documentada por [7]; sin embargo para la generación del datatest se prosiguió a partir de las corrupciones documentadas por [5]. Para el punto de comparación se utilizó un el datatest estándar de CIFAR-10.

**Tabla 3.** Comparación de resultados de pérdida y precisión de los modelos CNN obtenidos a través de fine tunning, con y sin entrenamiento.

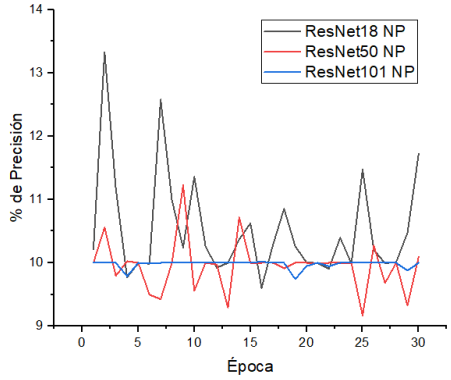
| Modelo | Datatest | Pretrain | Loss | Acc |
|---|---|---|---|---|
| ResNet18 | Cifar 10-C | No | 4426.91 | 13.28 % |
| | Cifar 10-C | Si | 780.97 | 11.24 % |
| | Cifar 10 | No | 2.91 | 31.2 % |
| | Cifar 10 | Si | 3.74 | 75.13 % |
| ResNet50 | Cifar 10-C | No | 7065.52 | 11.22 % |
| | Cifar 10-C | Si | 6799.6 | 14.57 % |
| | Cifar 10 | No | 17 | 26.87 % |
| | Cifar 10 | Si | 3.68 | 74.8 % |
| ResNet101 | Cifar 10-C | No | 3415399 | 10.01 % |
| | Cifar 10-C | Si | 1017160 | 10.78 % |
| | Cifar 10 | No | 19.22 | 20.31 % |
| | Cifar 10 | Si | 14.33 | 76.25 % |
| VGG16 | Cifar 10-C | No | 434.53 | 14.44 % |
| | Cifar 10-C | Si | 9047.64 | 12.91 % |
| | Cifar 10 | No | 2.52 | 22.12 % |
| | Cifar 10 | Si | 8.93 | 68.55 % |
| VGG19 | Cifar 10-C | No | 375.85 | 12.37 % |
| | Cifar 10-C | Si | 843.24 | 16.85 % |
| | Cifar 10 | No | 2.62 | 19.6 % |
| | Cifar 10 | Si | 9.32 | 68.46 % |
| ResNet50 | Cifar 10-C | No | 2.96 | 13.01 % |
| | Cifar 10-C | Si | 26.98 | 23.81 % |
| | Cifar 10 | No | 2.92 | 13.37 % |
| | Cifar 10 | Si | 11.44 | 64.98 % |

### 5.2. Modelos de arquitecturas

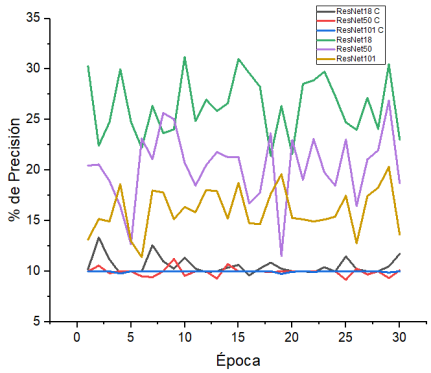
Como base se utilizó el principio de "Fine Tunning" siendo las arquitecturas evaluadas en este experimento: ResNet 18, 50 y 101 [4], así como los modelos VGG16, VGG19 [12] y AlexNet [8].

### 5.3. Configuraciones del modelo

Se adaptó el modelo de Fine Tunning [1] congelando los parámetros del modelo para que no se actualicen durante el entrenamiento, después se reemplazó la última capa de la red para que tuviera el mismo número de salidas que el número de clases en el conjunto

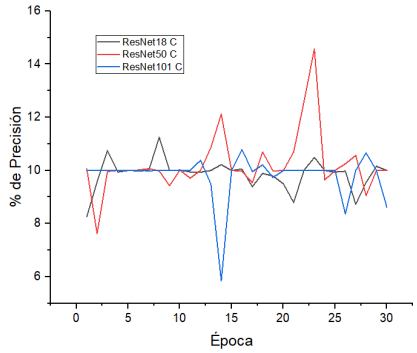



(a) Modelo ResNet18, 50 y 101 sin preentrenamiento (NP).

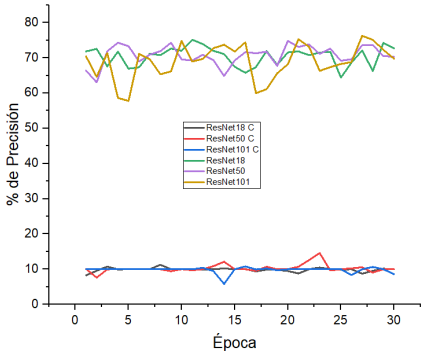



(b) % de precisión del modelo sin pesos de ImageNet del datatest corrupto (ResNetX-C) vs modelo sin pesos del datatest sin corrupciones.

**Fig. 6.** Porcentaje de precisión de la arquitectura ResNet.



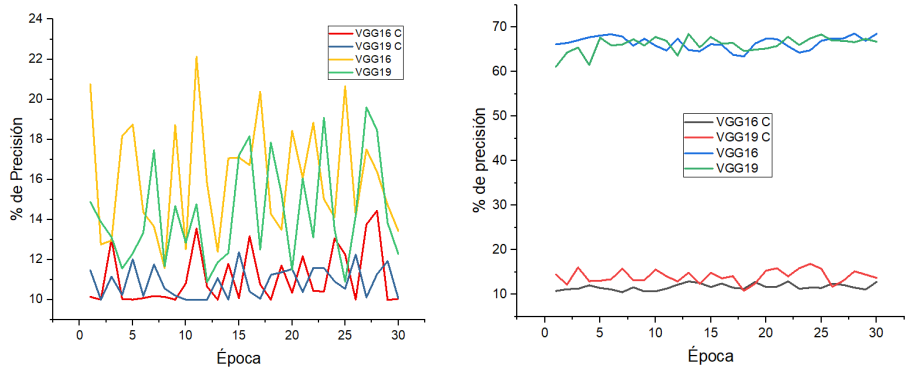

(a) Modelo ResNet18, 50 y 101.




(b) % de precisión del modelo con pesos de ImageNet del datatest corrupto (ResNetX-C) vs modelo del datatest sin corrupciones.

**Fig. 7.** Porcentaje de precisión de la arquitectura ResNet con preentrenamiento en ImgeNet.

de datos CIFAR10. Cada arquitectura se entrenó en primer término, sin pesos, con un dataset basado en cifar-10, para después evaluar el modelo con el dataset de imágenes corruptas por cifar-10-C, como se puede apreciar en la Fig. 5, evaluando la precisión y pérdida en cada una de las 30 épocas, con Adam como optimizador y Cross Entropy loss como criterion. En segundo término se utilizaron las mismas variables pero con el modelo preentrenado en ImageNet. En tercera instancia se realizaron ambos procesos con un datatest basado en Cifar-10 sin corrupciones, con el objetivo de analizar el gap y comportamiento de los modelos antes mencionados ante entradas con corrupciones naturales y sin ellas.




(a) % de precisión de la CNN VGG16 Y 19 evaluadas sin pesos de ImageNet con Corrupciones (C) y sin ellas.

(b) % de precisión de la CNN VGG16 Y 19 evaluadas cpn preentrenamiento de ImageNet con Corrupciones (C) y sin ellas.

**Fig. 8.** Porcentaje de precisión de la arquitectura VGG.

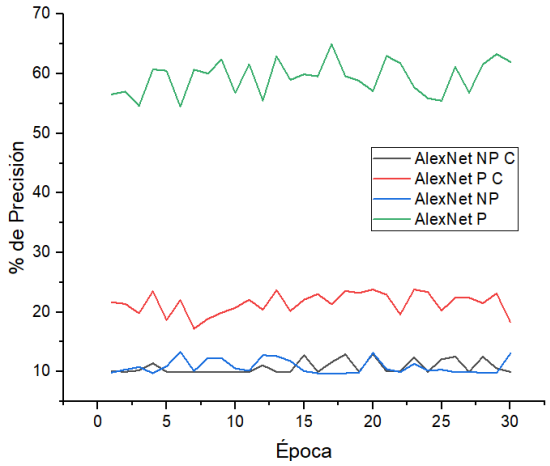



**Fig. 9.** Precisión obtendia del modelo AlexNet validado sin preeentrenamiento (NP) y con preentrenamiento, en datatest basado en Cifar10 corrupto (c) y sin corrupciones.

## 6. Resultados

Tras el análisis de las diferentes arquitecturas de redes neuronales, configuradas según lo antes mencionado, en primer lugar para los modelos sin preentrenar, resalta con los mejores resultados el modelo ResNet, en específico la ResNet18, con un porcentaje de precisión del 13.32 %, Figura 7,mientras que la ResNet101 tuvo el peor modelo para este caso, incluso cambiando los parámetros del learning rate, sin embargo, comparados con las validaciones del dataset sin corrupciones se ven claramente opacadas todas las arquitecturas, Fig. 6 inciso b.

Cuando esta arquitectura fue entrenada y evaluada con los pesos de ImageNet, se puede observar una mejoría en la precisión y un descenso en la pérdida, siendo ahora ResNet50 el que mejor responde ante perturbaciones, cuando estos resultados se comparan con sus homólogos sin corrupciones se aprecia que el modelo no tiene problemas, teniendo una precisión promedio del 75.39 %.

Para la Arquitectura VGG, evaluando el modelo sin preentrenamiento se puede observar, Fig.8, que no hay una gran diferencia entre la precisión obtenida con el datatest de CIFAR10-C y su homologo sin corrupciones; En contraste, cuando el modelo se evalúa con los pesos de ImageNet la diferencia entre el modelo con corrupciones es de 68.50 % (promedio) comparado contra el datatest sin ruido.

En la arquitectura AlexNet se puede apreciar, Fig. 9, que el comportamiento de la precisión es muy parecido sin preentrenamiento, tanto con un datatest con corrupciones y sin ellas, además demostró ser la mejor Red neuronal respecto a precisión, llegando a tener 23.81 % exitoso, con unos rangos de perdía muy bajos comparados con las otras arquitecturas.

## 7. Conclusiones y limitaciones

Con base en los resultados se puede afirmar que conforme a la literatura, los modelos de redes neuronales con menos capas y clases son más robustos antes ataques con corrupciones naturales, ya que al ser más generales los detalles y ruidos tienden a afectar en menor medida la respuesta de la Red Convolucional. Se puede apreciar un patrón de diferencia en las diferentes arquitecturas cuando las entradas tienen corrupciones de tipo natural, por ejemplo, el Modelo ResNet tiene una diferencia promedio del 55 % menor de precisión cuando recibe entradas con ruido natural, en un modelo sin preentrenamiento, y de 86 % menor cuando esta preentrenada comparada con la precisión esperada ante entradas normales, esto sin importar las capas, Tabla 3. Por lo que se podría deducir el impacto en esta métrica conociendo la diferencia de precisión en alguna de sus presentaciones. Cabe aclarar que esta suposición está basada en estas 3 arquitecturas, haría falta corroborar esta hipótesis en CNN más complejas como lo es RaWideResNet-70-16, WideResNet-70-16, entre otras, así como probar diferentes datasets corruptos, en estas mismas, por ejemplo CIFAR100-C, CIFAR10-P, CCC, etc.

## Referencias

1. Dhillon, P. S., Foster, D., Ungar, L.: Transfer learning using feature selection (2009) doi: 10.48550/arXiv.0905.4022
2. Drenkow, N., Sani, N., Shpitser, I., Unberath, M.: A systematic review of robustness in deep learning for computer vision: Mind the gap? (2021) doi: 10.48550/arXiv.2112.00639
3. Finn, C., Abbeel, P., Levine, S.: Model-agnostic meta-learning for fast adaptation of deep networks. In: Proceedings of the 34th International Conference on Machine Learning, vol. 70, pp. 1126–1135 (2017)
4. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770–778 (2016) doi: 10.1109/CVPR.2016.90

5. Hendrycks, D., Dietterich, T.: Benchmarking neural network robustness to common corruptions and perturbations. In: International Conference on Learning Representations, pp. 1–16 (2019) doi: 10.48550/arXiv.1903.12261
6. Ju, H., Li, D., Zhang, H. R.: Robust fine-tuning of deep neural networks with hessian-based generalization guarantees. In: Proceedings of Machine Learning Research, pp. 10431–10461 (2022)
7. Krizhevsky, A., Hinton, G.: Convolutional deep belief networks on cifar-10 (2010)
8. Krizhevsky, A., Sutskever, I., Hinton, G. E.: Imagenet classification with deep convolutional neural networks. Advances in Neural Information Processing Systems, vol. 25, pp. 1–9 (2012)
9. Pan, S. J., Yang, Q.: A survey on transfer learning. IEEE Transactions on knowledge and data engineering, vol. 22, no. 10, pp. 1345–1359 (2009) doi: 10.1109/TKDE.2009
10. Pratt, L., Jennings, B.: A survey of transfer between connectionist networks. Connection Science, vol. 8, no. 2, pp. 163–184 (1996) doi: 10.1080/095400996116866
11. Rice, J. R.: The algorithm selection problem. Advances in computers, vol. 15, pp. 65–118 (1976) doi: 10.1016/S0065-2458(08)60520-3
12. Shadeed, G. A., Tawfeeq, M. A., Mahmoud, S. M.: Automatic medical images segmentation based on deep learning networks. In: IOP Conference Series: Materials Science and Engineering, vol. 870, pp. 012117 (2020) doi: 10.1088/1757-899X/870/1/012117
13. Szegedy, C.: Intriguing properties of neural networks. In: Proceedings of the International Conference on Learning Representations, pp. 1–10 (2013)
14. Taori, R., Dave, A., Shankar, V., Carlini, N., Recht, B., Schmidt, L.: Measuring robustness to natural distribution shifts in image classification. Advances in Neural Information Processing Systems, vol. 33, pp. 18583–18599 (2020)
15. Wang, S., Veldhuis, R., Strisciuglio, N.: The robustness of computer vision models against common corruptions: A survey (2023) doi: 10.2139/ssrn.4960634
16. Xie, Q., Luong, M. T., Hovy, E., Le, Q. V.: Self-training with noisy student improves imagenet classification. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10687–10698 (2020) doi: 10.1109/CVPR42600.2020.01070