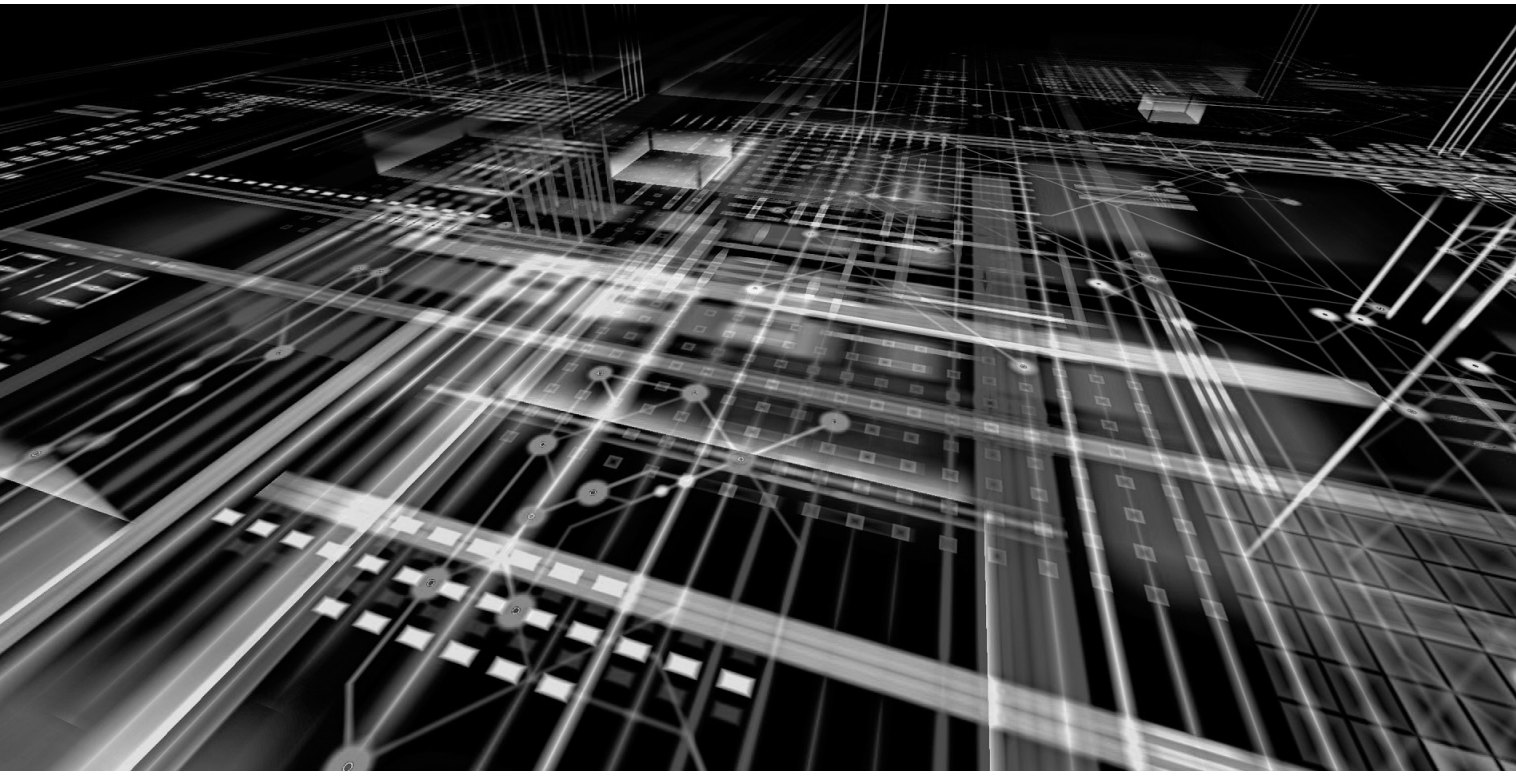
Técnicas de aumento de datos para imágenes aéreas y evaluación de rendimiento en modelos de *deep learning*

Cristian Camilo Gómez Sarasa[1]
José David Ortega Pabón[2]

Data augmentation techniques for aerial images and performance evaluation in deep learning models



1 Practicante de Ingeniería Electrónica, Grupo de Investigación Centro de Desarrollo Tecnológico Aeroespacial para la Defensa –CETAD– (Rionegro, Antioquia, Colombia). Correo electrónico: camilo.gomezsar@hotmail.com.

2 Magíster en TIC, Grupo de Investigación Centro de Desarrollo Tecnológico Aeroespacial para la Defensa –CETAD– (Rionegro, Antioquia, Colombia). Correo electrónico: jdavid_ortega@hotmail.com.

Cómo citar este artículo:
Gómez Sarasa, C. C. & Ortega Pabón, J. D. (2020). Técnicas de aumento de datos para imágenes aéreas y evaluación de rendimiento en modelos de deep learning. *Revista Universidad Católica de Oriente, 31*(45), 100-115.

Resumen

En la actualidad, uno de los principales problemas que afronta un profesional que trabaja con modelos de *deep learning* (DL) es la limitada información que existe para la implementación de estos modelos, por esta razón, se hace necesario el desarrollo de metodologías y técnicas que permitan mejorar el aprovechamiento de los datos con los que se cuenta; una de las principales metodologías para estos fines es el aumento de datos, que consiste en que a partir de un conjunto de datos inicial se modifican cada uno de los datos, de manera que cada una de estas transformaciones representen un dato nuevo para el conjunto. Por esta razón, en el presente artículo se exponen diversas técnicas de aumento de datos para imágenes aéreas y la evaluación de algunos de los principales modelos de *deep learning*. Tras ser entrenados con un conjunto de datos inicial, los resultados fueron contrastados con los resultados de los mismos modelos entrenados con un conjunto de datos aumentado, donde se pudo observar la mejoría en el desempeño de los modelos, disminuyendo el sobreajuste y aumentando la capacidad de generalización de estos modelos. Este trabajo se realizó bajo herramientas de *software* libre, con el uso de una computadora con sistema operativo Ubuntu 16.04, la programación de cada uno de los algoritmos bajo Python3 y para el despliegue se usó una tarjeta gráfica Nvidia Quadro P2200.

Palabras clave

Data augmentation, *deep learning*, procesamiento digital de imágenes, clasificación de imágenes, imágenes aéreas.

Abstract

At present, one of the main problems for a professional who works with Deep Learning (DL) models, is the limited amount of data that is available for the implementation of these models, for this reason, development is necessary of methodologies and techniques that improve the use of available data; one of the main methodologies for this purpose is the Data augmentation, which is that from an initial data set, transformations are made in each of the data, so that each of these transformations represents a new data for the data set. For this reason, this article presents several techniques to increase aerial image data and the evaluation of some of the main Deep Learning models, after being trained with an initial data set, the results were compared with the results of trained models with an increased data set (from the initial data set), where it was possible to observe the improvement in the performance of the models after being trained with the enlarged set of images, decreasing overfitting and increasing the ability to generalize these models; this work was carried out under free software tools, with the use of a computer with Ubuntu 16.04 operating system, the programming of each of the algorithms under Python3, for the deployment an Nvidia Quadro P2200 graphics card was used.

Key words

Data augmentation, deep learning, digital image processing, image classification, aerial images.

Intoducción

Con los avances en inteligencia artificial (IA), las redes neuronales convolucionales (CNN) se han convertido en una herramienta poderosa para el procesamiento y reconocimiento del contenido de las imágenes, gracias a avances como las unidades de procesamiento de gráfico (GPU) que permiten la implementación de modelos cada vez más complejos (hasta con cientos de millones de parámetros) (Wu, Yan, Shan, Dang y Sun, 2015).

Por otra parte, la clasificación y detección de objetos en imágenes aéreas ha evolucionado progresivamente, impulsada por aplicaciones en áreas de seguridad, vigilancia, búsqueda y rescate, agricultura (Zhang y Kovacs, 2012) y ganadería (Okafor, Smit, Schomaker y Wiering, 2017).

Así mismo, el extenso uso de tecnologías o dispositivos como aeronaves remotamente tripuladas y drones, ha incrementado las necesidades de algoritmos más robustos para la clasificación y detección de objetos en imágenes aéreas. Sin embargo, estas aplicaciones representan un desafío computacional, a causa de la gran variación entre imágenes, debido a deformaciones, tamaño o dimensiones de los objetos representados en pequeñas porciones de pixeles, haciendo la detección y clasificación más difícil (Lee, Wang, Crandall, Šabanović y Fox, 2017).

Por lo anterior, se hace necesario un amplio conjunto de datos para realizar el entrenamiento de los modelos deseados, de manera que se obtengan resultados aceptables, y a su vez, evitar también el sobreajuste de estos modelos.

En la actualidad existen diferentes algoritmos ampliamente usados para el reconocimiento del contenido de las imágenes. Algunos basados en procesamiento digital de imágenes (PDI), métodos estadísticos y métodos basados en IA. Algunos de los métodos propuestos son el reconocimiento a partir de la implementación de algoritmos como el histograma de orientación del gradiente (HOG) (Dalal y Triggs, 2005), el algoritmo *scale-invariant feature transform* (SIFT) (Lowe, 1999), el algoritmo *speeded-up robust features* (SURF) (Bay, Tuytelaars y Van Gool, 2006) y también modelos basados en CNN que han demostrado buen desempeño en los diferentes desafíos del conjunto de datos ImageNet (Deng, Dong, Socher, Li, Li y Fei-Fei, 2009) como son: VGG16, ResNet, Inception, MobileNet.

Si bien los modelos de DL han mostrado buenos resultados para clasificación y detección de objetos, para su desarrollo requieren de unas capacidades de cómputo elevadas, debido a la cantidad de datos con los que debe trabajar, la cantidad

de parámetros que cada uno de estos modelos implementa y la constante necesidad de que las aplicaciones donde se usan este tipo de modelos sean cada vez más precisas.

Es de resaltar que la eficiencia de estos modelos está estrechamente relacionada con el tipo de aplicaciones que se desee implementar, puesto que algunos de estos requieren decenas de millones de parámetros paras u conformación, otros más extensos implementan cientos de millones, para los que se requiere un coste de recursos computacionales mayor y, por ende, un mayor tiempo de inferencia por cada lote de imágenes que recibe el modelo (Figura 1); también es importante resaltar que no necesariamente los modelos con mayor número de parámetros entregan los mejores resultados (Figura 2).

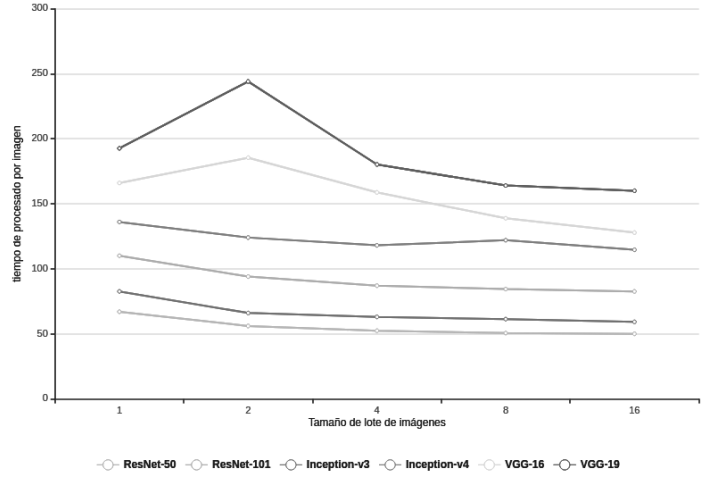


Figura 1. Tiempo de inferencia vs tamaño del lote de imágenes.

Fuente: creación propia, basado en los resultados expuestos en: Yalniz et al. *(2019); He* et al. *(2016); Singh* et al. *(2019); Szegedy* et al. *(2017) y Simonyan (2014).*

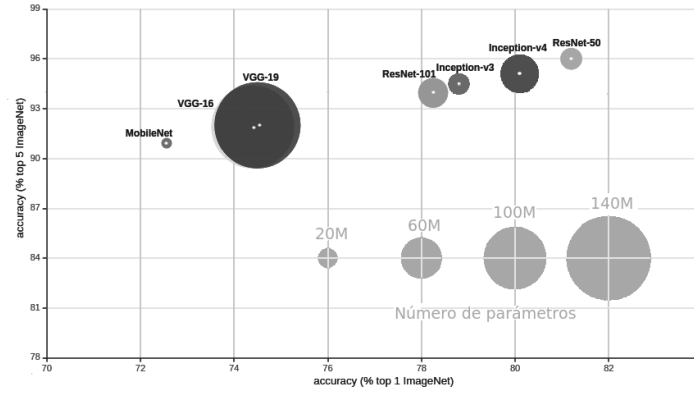


Figura 2. Diagrama de burbujas, accuracy top 1 ImageNet (eje x) vs accuracy top 5 ImageNet (eje y) vs cantidad de parámetros del modelo, tamaño de la burbuja especificado en «Número de parámetros».

Fuente: creación propia, basado en los resultados expuestos en: Yalniz et al. *(2019); He* et al. *(2016); Singh* et al. *(2019); Szegedy* et al. *(2017); Simonyan (2014); y Howard* et al. *(2017).*

Aumento de datos

Una de las principales limitantes para la implementación de modelos de DL es la creación de conjuntos de datos representativos que contengan objetos de interés para su área de trabajo. Esta limitación puede llevar a un sobreajuste del modelo viéndose reflejado en resultados erróneos en las predicciones al ingresar nuevos datos.

Por esta razón, se hace necesaria la implementación de métodos que permitan aumentar el desempeño de los modelos haciendo un mejor uso de los datos con los que se cuenta. Uno de los métodos más usados para estos fines es el aumento de datos.

El aumento de datos es una técnica ampliamente usada en el ámbito de la IA, cuya finalidad es aumentar el conjunto de datos aplicando diversas transformaciones al conjunto original, teniendo en cuenta que los nuevos datos obtenidos sean representativos y diferentes para los modelos a entrenar. Con esto se busca el mejoramiento de las métricas de desempeño (evitando así el sobreajuste) para obtener mejores resultados de clasificación.

Existen diferentes tipos de estrategias de aumento de datos para la implementación de modelos de DL. Algunas de estas transformaciones son: rotaciones, recortes, desplazamientos y conversiones geométricas, que se realizan a las imágenes y que permiten la generación de otra nueva, que, para el modelo a implementar, representa información diferente. Todo esto bajo la premisa de «cuantos más datos tenga acceso un algoritmo de ML, más eficaz puede ser » (Pérez y Wang, 2017).

Uno de los aspectos que se debe tener en cuenta a la hora de realizar este proceso de aumento de datos es el tipo de aplicación con el que se trabaja (biomédicas, satelitales, aéreas, Street View, etc.) (Buslaev, Parinov, Khvedchenya, Iglovikov y Kalinin, 2018); por esta razón es conveniente realizar las transformaciones de acuerdo con el desarrollo específico y la variabilidad entre las imágenes, puesto que algunas transformaciones no son representativas para la aplicación en particular de cada modelo. Por ejemplo, para aplicaciones aéreas y satelitales puede ser conveniente para el aumento de los datos realizar una transformación basada en reflexión vertical a una imagen de un automóvil (Figura 3 (a)), pues con esta transformación se puede crear un nuevo dato que para el modelo representa un dato «diferente» (Figura 3 (b)) y es representativo para él. Por otro lado, una transformación de este tipo para aplicaciones de Street View, aunque se muestra como un nuevo dato para el modelo, este no es representativo, pues con esta transformación, el automóvil en la imagen quedaría al revés (Figura 3 (c), (d)), no siendo coherente con el contexto.

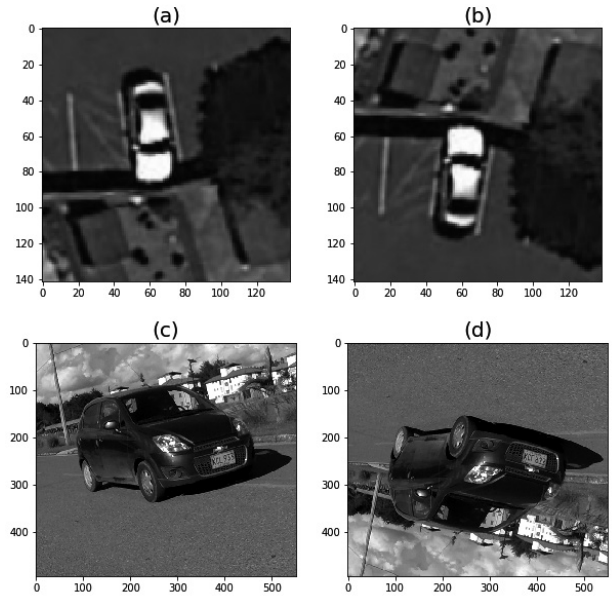


Figura 3. Reflexión vertical, aplicada a dos imágenes de automóvil: (a) imagen aérea de un automóvil; (b) imagen aérea de un automóvil reflejada verticalmente; (c) imagen vertical de un automóvil; (d) imagen vertical de un automóvil reflejada verticalmente.

Fuente: Elaboración propia.

Metodología

A continuación, se exponen cada uno de los pasos realizados para el desarrollo de este trabajo, desde la obtención del conjunto de imágenes hasta la implementación de los modelos de prueba:

Selección de modelos: la elección de los modelos para el desarrollo de este trabajo se realizó basada en el desempeño obtenido por cada uno de estos en los diferentes desafíos ImageNet (Russakovsky, Deng, Su, Krause, Satheesh, Ma, ... y Berg, 2015), así como en la cantidad de parámetros y el tiempo de inferencia mostrado por cada uno de estos modelos (Figura 1 y Figura 2). Por esta razón los modelos elegidos para la evaluación son:

- VGG16
- InceptionV4
- MobileNet

Conjunto de imágenes: para el desarrollo de este trabajo se implementó un set de imágenes basado en el *conjunto de datos a gran escala para la detección de objetos en imágenes aéreas* (DOTA) (Xia, Bai, Ding, Zhu, Belongie, Luo, ... y Zhang, 2018) el cual contiene 2806 imágenes aéreas pertenecientes a 15 clases, con tamaño de hasta pixeles, con múltiples instancias de cada clase.

Para efectos de este ejercicio se realizaron recortes de una porción de este set de imágenes, con tamaños cercanos a 100 × 100 pixeles (con una única instancia de cada objeto), eligiendo solo

aquellas que contenían instancias de las clases: avión, barco, automóvil y helicóptero, obteniendo de esta forma 2000 imágenes, distribuidas en 500 imágenes por clase. Posterior a esto, se realizó el proceso de aumento de datos partiendo de una imagen original (Figura 4) y realizando las transformaciones respectivas (Figura 5).

Aumento de imágenes: inicialmente, se separó el set de imágenes en dos conjuntos: uno de estos para entrenamiento, para el cual se seleccionaron 400 imágenes por cada clase para un total de 1600, y el otro conjunto de datos para validación; para este se seleccionaron 100 imágenes por clase para un total de 400.

Para el aumento del set de imágenes se seleccionaron seis transformaciones, para generar seis imágenes adicionales por cada imagen del conjunto de datos de entrenamiento (1600 imágenes), obteniendo así un total de 11 200 imágenes para el entrenamiento de los modelos elegidos, manteniendo el conjunto de datos para validación. Las transformaciones que se eligieron fueron:

- Rotaciones 90°, 180°, 270° (Figura 5 (a, b, c)): esta transformación consiste en el desplazamiento angular de todos los pixeles tomando como punto central $\left(\frac{w}{2}, \frac{h}{2}\right)$. siendo *w* el ancho y *h* el alto de la imagen.
- Transformación afín (Figura 5 (d)): la transformación afín matemáticamente se basa en una transformación lineal (multiplicación matricial) y una traslación (adición vectorial), conservando propiedades como colinealidad, es decir, todos los puntos de una recta en su espacio original se preservan en el nuevo espacio generado. Para realizar esta transformación a una imagen es necesario definir dos conjuntos de tres puntos, donde el primer conjunto corresponden a los vértices de un triángulo en la imagen de origen, los segundos tres puntos representan la nueva ubicación de los primeros tres puntos, y a partir de estos se realizan los desplazamientos respectivos de cada uno de los pixeles en una imagen. Para este caso particular los dos grupos de puntos corresponden a

$$\left[\left(\frac{w}{4}, \frac{h}{4}\right), \left(\frac{w}{4}, \frac{3h}{4}\right), \left(\frac{3w}{4}, \frac{h}{4}\right)\right] \quad \text{y} \quad \left[\left(\frac{w}{4}, \left(\frac{h}{4} - \frac{h}{6}\right)\right), \left(\left(\frac{w}{4} + \frac{w}{6}\right), \left(\frac{3h}{4} - \frac{h}{8}\right)\right), \left(\frac{3w}{4}, \left(\frac{h}{4} + \frac{h}{6}\right)\right)\right]$$

respectivamente.

- Reflexión horizontal (Figura 5 (e)): consiste en la simetría axial de todos los pixeles tomando como eje de simetría la línea vertical que atraviesa el pixel $\left(\frac{w}{2}, \frac{h}{2}\right)$.
- *Zoom out* (Figura 5 (f)): consiste en el reescalado de la imagen a un tamaño menor que el original. Para efectos de este trabajo se realizó un reescalado de cada imagen en un 50 %.

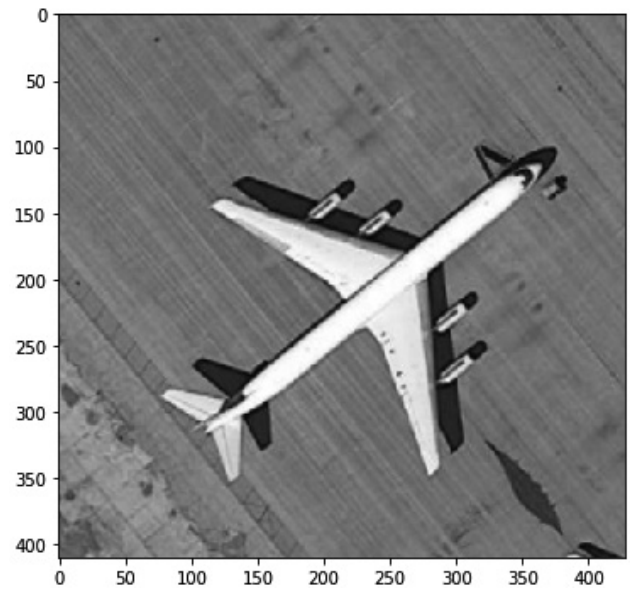


Figura 4. Imagen aérea original a la que se le realizan las transformaciones seleccionadas.

Fuente: Elaboración propia.

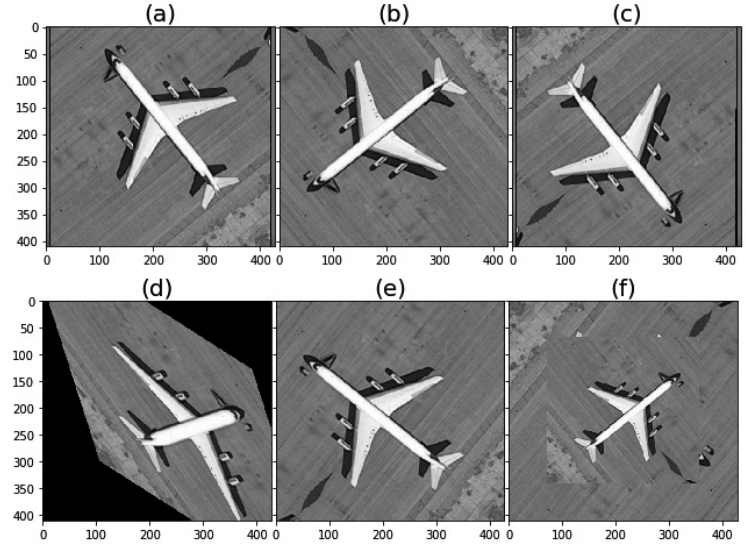


Figura 5. Imagen aérea original a la que se le realizan las transformaciones seleccionadas.

Fuente: Elaboración propia.

Implementación de los modelos: para la implementación de los modelos, inicialmente se realizó el redimensionamiento de las imágenes a un tamaño estándar de 100×100 pixeles. Luego se procedió con el etiquetado del grupo de imágenes, creando un arreglo de matrices para cada uno de los conjuntos de imágenes (entrenamiento y validación) y generando un vector de etiquetas correspondientes a cada imagen en su respectiva posición en el arreglo de matrices. Posteriormente se configuraron los parámetros necesarios para la implementación de los modelos.

Para lo anterior, se adaptó la capa superior de los modelos de manera que el parámetro de entrada admitiese las imágenes del conjunto de datos cuyo tamaño es de 100 × 100 pixeles. Las capas inferiores (o de salida) se modificaron para que solo se consideraran las cuatro clases seleccionadas, (como se muestra en la Figura 6), congelando las capas que los modelos traen por defecto y agregando las capas personalizadas, para esto, se adicionó una capa *flatten* (aplanamiento de las matrices luego de las capas convolucionales convirtiéndolas en un vector de datos), una capa densa (o capa completamente conectada que se encarga del procesamiento de los vectores de datos, para determinar la clase a la que el objeto pertenece) con función de activación ReLu, una capa de *dropout* (capa que se encarga de desconectar una porción de neuronas entre la capa densa y la capa de clasificación y de esta manera evitar el sobreajuste de los modelos) configurada con un valor de 0,4, y por último, una capa densa para clasificación, adaptándola a las cuatro clases con las que se trabajó. Luego se procedió con la configuración para entrenamiento, seleccionando como función de pérdida entropía cruzada y como método optimizador RMSprop (Bengio, 2015). Finalmente, se procedió con el entrenamiento de cada uno de los modelos seleccionados y la evaluación de resultados.

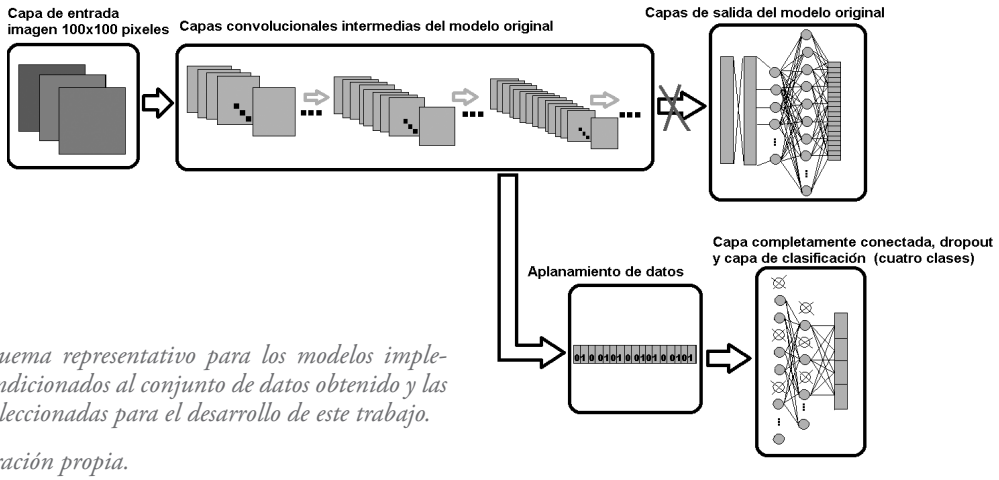


Figura 6. Esquema representativo para los modelos implementados, acondicionados al conjunto de datos obtenido y las cuatro clases seleccionadas para el desarrollo de este trabajo.

Fuente: Elaboración propia.

Resultados

Luego de la configuración de cada uno de los modelos se procedió con el entrenamiento en una computadora con un sistema operativo Ubuntu 16.04, con tarjeta gráfica Nvidia Quadro P2200, durante 100 épocas y realizando finalmente la validación del modelo. Este proceso se hace tanto para los datos originales (sin realizar proceso de aumento de datos), como para las 11 200 imágenes obtenidas luego del aumento de datos.

La metodología con la que se realizó el análisis del rendimiento fue a través de la evaluación de la exactitud (*accuracy*) y la pérdida (*loss*), tanto en entrenamiento como en validación.

En la Figura 7 se puede observar el comportamiento a lo largo del entrenamiento y validación de cada uno de los modelos sin realizar ningún tipo de aumento de datos. Se puede evidenciar de acuerdo con los gráficos (a) y (b), que los resultados obtenidos fueron considerablemente buenos para el entrenamiento, obteniendo una exactitud superior al 90 % en cada uno de los modelos (desde las épocas iniciales) y pérdidas cercanas a cero. Sin embargo, en los gráficos (b) y (c) correspondientes a la exactitud y a la pérdida de los modelos en validación respectivamente, se observan constantes fluctuaciones, causando errores en las predicciones. También se puede observar una diferencia considerable entre las pérdidas en entrenamiento y en validación (como se muestra en la Tabla 1), dando indicios de que los modelos tenían tendencias al sobreajuste para el set de datos original y las imágenes para validación.

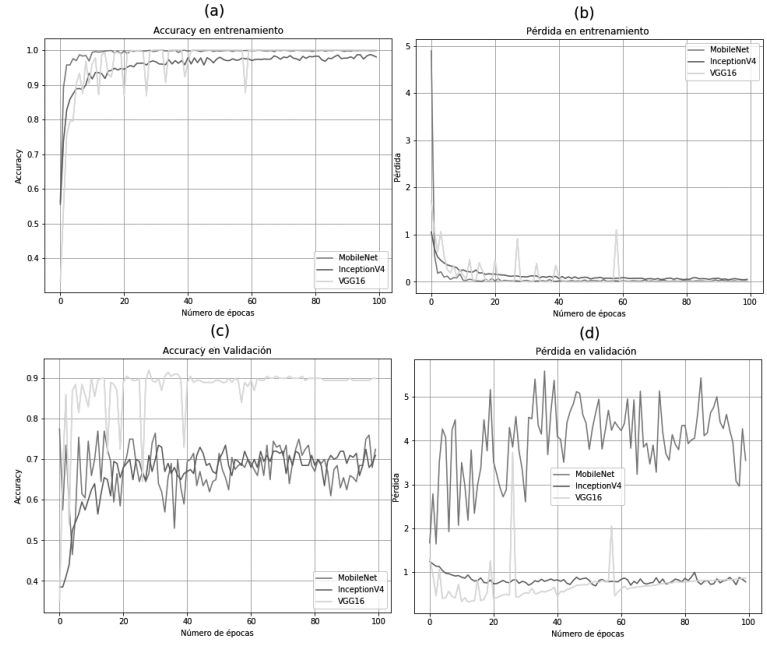


Figura 7. Resultados de los modelos en entrenamiento y validación sin realizar aumento de datos. Figura 7 (a). Número de épocas vs accuracy en entrenamiento; Figura 7 (b). Número de épocas vs pérdida del modelo por época en entrenamiento; Figura 7 (c). Número de épocas vs accuracy en validación; Figura 7 (b). Número de épocas vs pérdida del modelo por época en validación.

Fuente: Elaboración propia.

En la Figura 8 se observan los resultados obtenidos para el conjunto de imágenes con aumento de datos, donde se evidencia una uniformidad tanto para el entrenamiento ((a) y (b)) como para la validación ((c) y (d)), sin fluctuaciones notables y una mejora considerable frente a los resultados en validación, respecto a los resultados de los modelos entrenados sin aumento de datos.

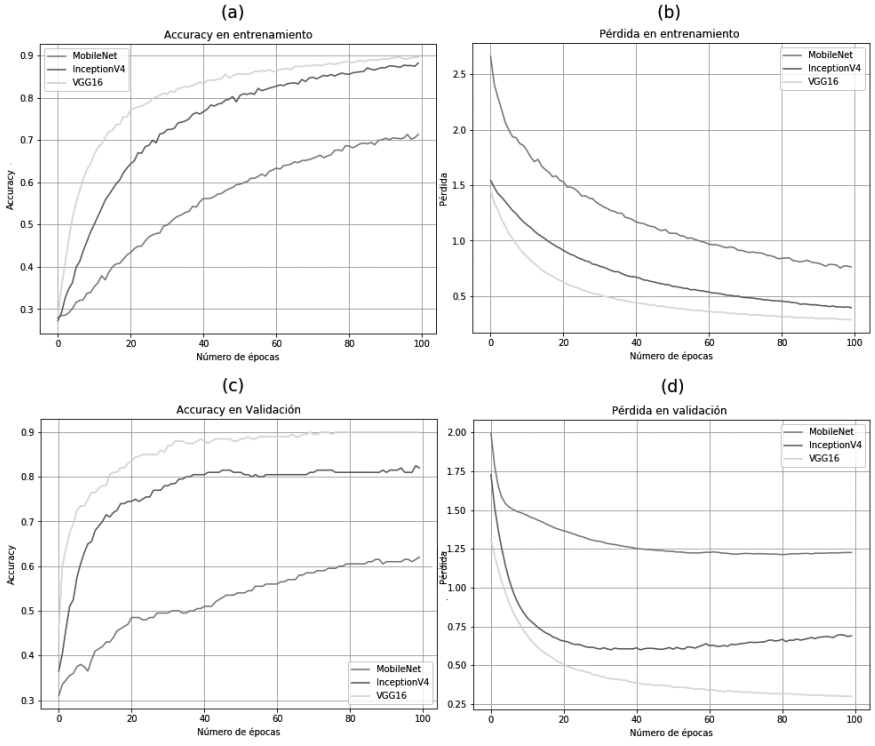


Figura 8. Resultados de los modelos en entrenamiento y validación tras realizar aumento de datos. Figura 8 (a). Número de épocas vs accuracy en entrenamiento; Figura 8 (b). Número de épocas vs pérdida del modelo por época en entrenamiento; Figura 8 (c). Número de épocas vs accuracy en validación; Figura 8 (b). Número de épocas vs pérdida del modelo por época en validación).

Fuente: Elaboración propia.

La Tabla 1 muestra los resultados finales tras el entrenamiento y validación de los modelos seleccionados, donde se puede observar una mejora y uniformidad en los resultados para entrenamiento y validación con valores muy equilibrados.

Tabla 1. Resultados de entrenamiento y validación para los modelos seleccionados para los conjuntos de datos sin aumento y con aumento de datos tras 100 épocas.

	Sin aumento de datos				Con aumento de datos			
	Accuracy		Pérdida		Accuracy		Pérdida	
	Entrenamiento	Validación	Entrenamiento	Validación	Entrenamiento	Validación	Entrenamiento	Validación
MobileNet	100,00%	71,88%	0,0197	3,675	72,30%	62,60%	0,74	1,223
InceptionV4	97,20%	70,96%	0,0756	0,894	89,40%	82,70%	0,368	0,668
VGG16	100,00%	89,50%	0,0000028	0,913	89,70%	90,00%	0,243	0,256

Fuente: Elaboración propia.

Conclusiones

De acuerdo con los resultados obtenidos, se infiere que los modelos que fueron implementados con el set de datos original que contenía 1600 imágenes de entrenamiento, exponían una tendencia al sobreajuste, evidenciado en los gráficos de pérdida, donde se puede observar una clara diferencia entre la pérdida para entrenamiento y la pérdida para validación. Por esta razón, se debe resaltar la importancia del manejo de las imágenes y la obtención de un buen conjunto de datos para lograr un desempeño óptimo de los modelos que se implementen.

De los tres modelos que se implementaron, el que mostró mejor desempeño tanto en entrenamiento como en validación fue el VGG16, para ambos conjuntos de datos, especialmente para el conjunto de datos aumentado, donde la diferencia tanto en entrenamiento como en validación era considerablemente pequeña, manteniéndose esta tendencia en los gráficos de pérdida, donde se puede intuir una buena capacidad de generalización. Cabe aclarar que para los demás modelos se obtuvieron resultados aceptables en cuanto al conjunto de datos aumentado, en contraste con los modelos implementados para el conjunto de datos original.

Aunque los resultados mostrados del modelo MobileNet fueron relativamente menores en comparación con los modelos VGG16 e Inception, cabe resaltar que este modelo es mucho más liviano en cuanto a la cantidad de parámetros, por lo que en términos generales los resultados obtenidos fueron buenos (observándose una clara mejoría en comparación con los resultados arrojados para este modelo sin realizar el aumento de los datos), en comparación con la cantidad de parámetros que se implementan para este modelo, por lo que se hace recomendable para aplicaciones en las que no se cuenta con *hardware* acelerador.

En términos generales, los resultados expuestos mostraron un mejor del desempeño y una mejor capacidad de generalización de los datos para los modelos que fueron entrenados con el conjunto de imágenes aumentado; por esta razón, como trabajos futuros se plantea la implementación de modelos de detección de objetos en imágenes aéreas para un conjunto de imágenes mayor, aplicando la técnica de aumento de datos.

Referencias bibliográficas

Bay, H., Tuytelaars, T. y Van Gool, L. (2006). Surf: Speeded up robust features. En *European Conference on Computer Vision* (pp. 404-417). Berlín: Springer.

Bengio, Y. (2015). Rmsprop and equilibrated adaptive learning rates for non-convex optimization. *corr abs/1502.04390*.

Buslaev, A., Parinov, A., Khvedchenya, E., Iglovikov, V. I. y Kalinin, A. A. (2018). Albumentations: fast and flexible image augmentations. *arXiv preprint arXiv:1809.06839*.

Dalal, N. y Triggs, B. (2005, June). Histograms of oriented gradients for human detection. En *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)* (Vol. 1, pp. 886-893). IEEE.

Deng, J., Dong, W., Socher, R., Li, L. J., Li, K. y Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. En *2009 IEEE Conference on Computer Vision and Pattern Recognition* (pp. 248-255). IEEE.

He, K., Zhang, X., Ren, S. y Sun, J. (2016). Deep residual learning for image recognition. En *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).

He, X., Cheng, K., Chen, Q., Hu, Q., Wang, P. y Cheng, J. (2019). Compact global descriptor for neural networks. *arXiv preprint arXiv:1907.09665*.

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., ... y Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*.

Lee, J., Wang, J., Crandall, D., Šabanović, S. y Fox, G. (2017). Real-time, cloud-based object detection for unmanned aerial vehicles. En *2017 First IEEE International Conference on Robotic Computing (IRC)* (pp. 36-43). IEEE.

Lowe, D. G. (1999). Object recognition from local scale-invariant features. En *Proceedings of the Seventh IEEE International Conference on Computer Vision* (Vol. 2, pp. 1150-1157). IEEE.

Okafor, E., Smit, R., Schomaker, L. y Wiering, M. (2017). Operational data augmentation in classifying single aerial images of animals. En *2017 IEEE International Conference on INnovations in Intelligent SysTems and Applications (INISTA)* (pp. 354-360). IEEE.

Perez, L. y Wang, J. (2017). The effectiveness of data augmentation in image classification using deep learning. *arXiv preprint arXiv:1712.04621*.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., ... y Berg, A. C. (2015). Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, *115*(3), 211-252.

Simonyan, K. y Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Singh, S. y Krishnan, S. (2019). Filter Response Normalization Layer: Eliminating Batch Dependence in the Training of Deep Neural Networks. *arXiv preprint arXiv:1911.09737*.

Szegedy, C., Ioffe, S., Vanhoucke, V. y Alemi, A. A. (2017). Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-first AAAI Conference on Artificial Intelligence*.

Wu, R., Yan, S., Shan, Y., Dang, Q. y Sun, G. (2015). Deep image: Scaling up image recognition. *arXiv preprint arXiv:1501.02876*, *7*(8).

Xia, G. S., Bai, X., Ding, J., Zhu, Z., Belongie, S., Luo, J., ... y Zhang, L. (2018). DOTA: A large-scale dataset for object detection in aerial images. En *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 3974-3983).

Yalniz, I. Z., Jégou, H., Chen, K., Paluri, M. y Mahajan, D. (2019). Billion-scale semi-supervised learning for image classification. *arXiv preprint arXiv:1905.00546*.

Zhang, C. y Kovacs, J. M. (2012). The application of small unmanned aerial systems for precision agriculture: a review. *Precision Agriculture*, *13*(6), 693-712.