Capítulo 4

Entrenamiento

Resumen: Utilizando los conjuntos de imágenes reales y sintéticas obtenidos, se procede al entrenamiento necesario para dotar a la aplicación de las capacidad de reconocimiento de imágenes. Para elegir la mejor relación proporcional entre reales y sintéticas se realizan una serie de ensayos, cuyos resultados se comparan y estudian para seleccionar la mejor relación entre facilidad de obtención del *dataset* y rendimiento de la red.

4.1. Entrenamiento

{**TODO TODO TODO:** While thousands of poses are required for pose estimation, [18], much less poses are necessary for object detection}

La fase de entrenamiento del proyecto está basada en las recomendaciones para generar modelos de TensorFlow Lite, ya que el modelo que se va a utilizar para la identificación es el de este formato. Para llevar a cabo el entrenamiento de la red neuronal se ha realizado un *script* que se encarga de la lógica de esto. Este *script* está basado en uno de los ejemplos de Tensorflow Lite y puede usarse desde Colaboratory[1], plataforma de Google que permite ejecutar y programar en Python desde el navegador. El ejemplo elegido como referencia consiste en identificación de flores[2], se ha seleccionado este debido a su similitud con el proyecto respecto a la identificación del objeto principal en un imagen.

Para generar el modelo usando las imágenes obtenidas se utiliza la biblioteca Model Maker de TensorFlow Lite, que permite llevarlo a cabo de manera sencilla mediante unas pocas líneas de código. El entrenamiento para este

[1] `https://colab.research.google.com/notebooks/intro.ipynb?utm_source=scs-index`

[2] `https://colab.research.google.com/drive/1sqBewUnvdATOO-yblj55EBFb2sM24XHR?hl=es-419`

proyecto se lleva a cabo desde la computadora, evitando tener que cargar todo el conjunto de imágenes en Colaboratory, para ello se tiene un *script*, similar al del ejemplo, que realiza la carga de las imágenes, el entrenamiento, la creación del modelo y las pruebas de rendimiento.

En el entrenamiento de las redes neuronales, generalmente se utiliza la mayor parte de los datos etiquetados para el entrenamiento en sí y los restantes de prueba, para medir la precisión de la red. Puesto que el código tomado de ejemplo está planteado para datasets formados únicamente por imágenes reales, la carga de estas se hace en conjunto y son separadas posteriormente en un 90 % para entrenamiento y el 10 % restante para *test*.

En cambio, en este proyecto, puesto que el entrenamiento se va a realizar a partir de imágenes generadas y reales mezcladas, y posteriormente se va a utilizar sobre objetos reales, para comprobar la viabilidad del modelo es necesario que las imágenes de *test* sean capturas reales solamente. Por lo tanto, en este caso, es necesario cargar las imágenes de entrenamiento y test ya separadas. Para dicha distribución se ha mantenido aproximadamente la proporción de 90 % y 10 % respectivamente. Con esto, la estructura de carpetas frente al entrenamiento queda como se muestra en la figura 4.1. Son necesarias dos carpetas principales, una en la que se guardan los datos de entrenamiento y en la otra los de test. Al tratarse de una red con aprendizaje supervisado, los datos deben ir correctamente etiquetados. Esto se traduce en que ambas carpetas contienen tres subcarpetas, una para cada material identificable (plástico, vidrio y metal), en las que se guardan las imágenes correspondientes. El nombre de estas subcarpetas será lo que se tome como etiquetas para la red.

El entrenamiento se divide en varios episodios. Estos corresponden al número de veces que el algoritmo recorre todos los datos. En cada episodio los datos se dividen en lotes, y en cada lote se recoge una pequeña parte de los datos. Cada lote se recorre en una iteración, por lo que cada episodio tiene las iteraciones necesarias para recorrer al completo el conjunto de datos. Esto significa que el número de iteraciones de cada episodio corresponde al resultado de dividir la cantidad de datos entre el tamaño de lote [55]. Por ejemplo, si se tienen 2400 imágenes y el tamaño de lote es 32, en cada episodio habría 75 iteraciones. Para estos valores se han mantenido los de por defecto de TensorFlow ya que se ha considerado que eran adecuados; siendo el número de episodios 5 y el tamaño de lote 32.

Con todo lo mencionado se genera el modelo entrenado con extensión *.tflite*, además de un documento de texto plano con las etiquetas de los distintos materiales. Estos dos archivos son lo que se han de trasladar a la aplicación móvil para ser utilizados.

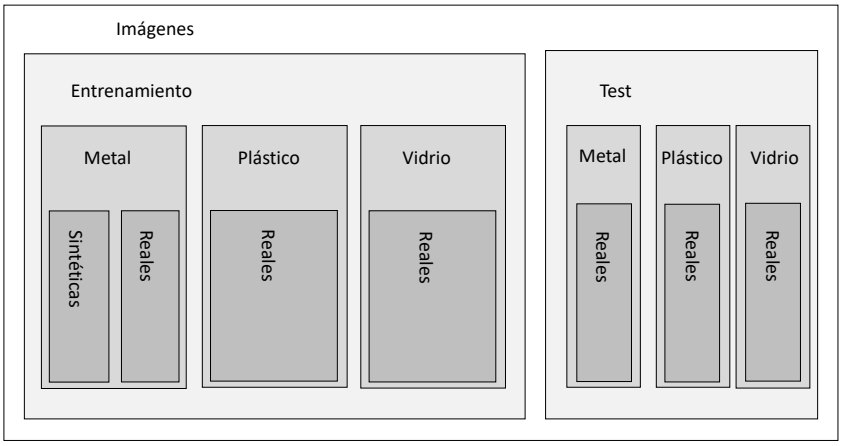

Figura 4.1: Organización de las carpetas con las imágenes separadas.

4.2. Resultados

Como último punto del entrenamiento, queda investigar con qué porcentaje de imágenes sintéticas se obtiene el mejor resultado en precisión que además el correcto funcionamiento de la aplicación.

Esta investigación se ha llevado a cabo con el *dataset* obtenido para el prototipado (explicado en la sección 3.2.2) de la aplicación. Esto supone que sólo uno de los tres materiales (metal) cuenta con imágenes sintéticas y reales mezcladas. En cambio, los dos materiales restantes (vidrio y plástico) están formados por completo por imágenes reales.

Para la comparación se ha comenzando con sólo imágenes reales, posteriormente se han ido aumentando paulatinamente el número de imágenes generadas en un 10 % hasta contar finalmente con un *dataset* de sólo imágenes sintéticas. La figura 4.2 corresponde al crecimiento de la precisión de la red durante el entrenamiento. La precisión es el porcentaje de aciertos de la red respecto a los resultados reales. Se encuentra representado para los distintas combinaciones de imágenes generadas y reales. En todos los casos crece de manera similar sin haber grandes diferencias según la proporción de imágenes sintéticas.

En la figura 4.3, por el contrario, se observa cómo varía la precisión al probarse el modelo con los datos de *test*, aquí se sacan las primeras conclu-

siones sobre la red. Se observa que la mayoría de las opciones se mantienen casi todo el proceso en una precisión mayor del 90 %. Los casos en los que se puede contemplar una caída de la precisión, son aquellos en los que más del 80 % de las imágenes son sintéticas.

La figura 4.4 muestra el valor final de la precisión en los distintos casos. A partir de esta última figura pueden sacarse las conclusiones de con qué porcentaje se obtendrían los mejores resultados.

Los dos *datasets* que cuentan con la mayor cantidad de imágenes sintéticas serían los menos recomendados para utilizar debido a su baja precisión. En cambio, entre el 0 % y el 70 % de imágenes generadas, se observa que la precisión siempre está por encima del 90 %, aunque nunca llega a superar el 95 %.

Entrenando la red únicamente con los materiales de vidrio y plástico, cuyas imágenes son todas reales, se obtiene un 96 % de precisión. Al comparar esto y el resultado del entrenamiento de la red con solamente imágenes reales, cuya precisióm , o con

Con los resultados obtenidos se considera como mejor opción utilizar un 70 % de imágenes generadas para el *dataset* final. Esta opción es de las que mayor precisión tienen y a la vez permite tener un *dataset* que funciona adecuadamente formado principalmente por imágenes sintéticas, lo que facilita la obtención de este.

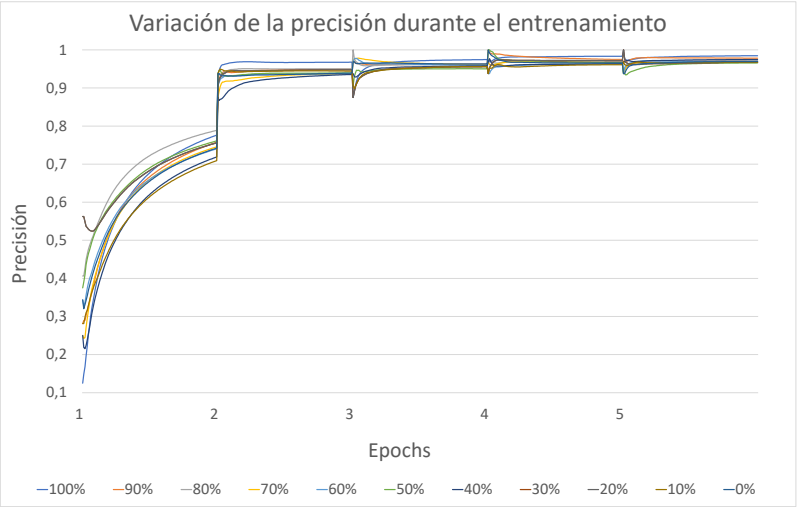

Figura 4.2: Variación de la precisión de la red neuronal a lo largo del entrenamiento de esta.

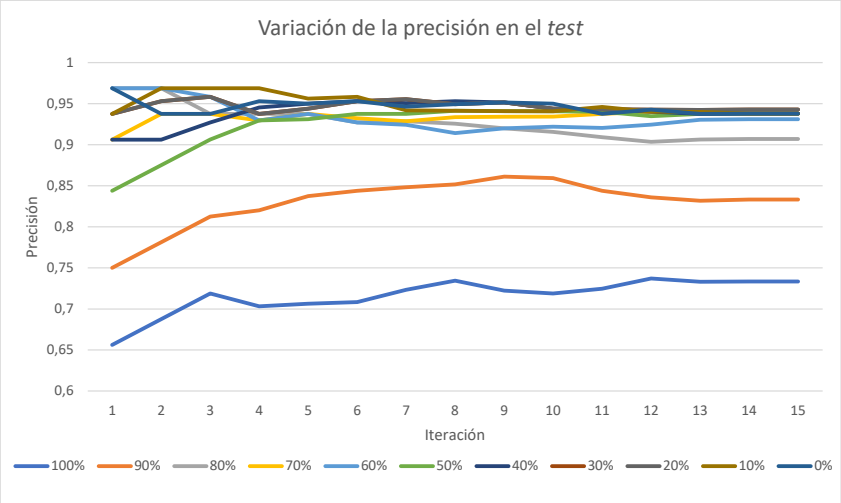

Figura 4.3: Variación de la precisión de la red neuronal durante las pruebas de esta.

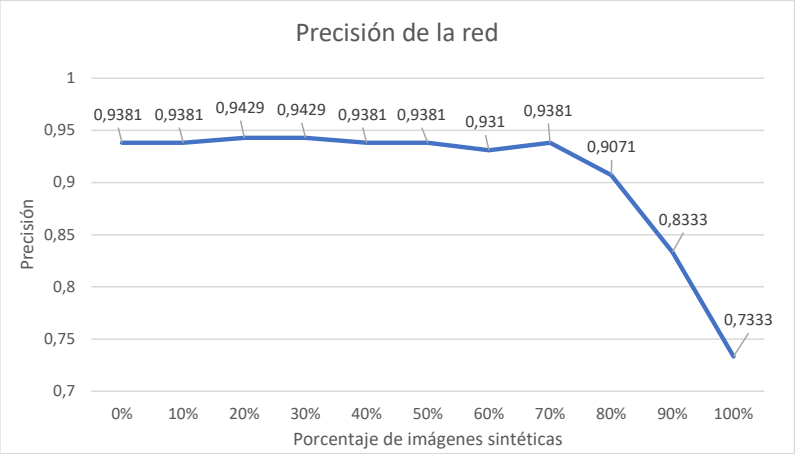

Figura 4.4: Variación de la precisión de la red neuronal según la proporción de imágenes reales y sintéticas.

Bibliografía

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] C. Alfonso, R. Estévez Estévez, J. M. Lobo, B. Lozano Diéguez, F. Prieto, J. Santamarta, and A. Gaerter. Emergencia climática en España. Diciembre 2016.

[3] R. Almond, G. M., and T. Petersen. Wwf (2020) living planet report 2020 - bending the curve of biodiversity loss. *World Wildlife Fund (WWF)*, 2020.

[4] Y. Amit. *2D Object Detection and Recognition: Models, Algorithms, and Networks*. Mit Press. MIT Press, 2002.

[5] Y. Amit, P. Felzenszwalb, and R. Girshick. *Object Detection*. Springer International Publishing, Cham, 2020.

[6] X. Basogain Olabe. Redes neuronales artificiales y sus aplicaciones. *Dpto. Ingeniería de Sistemas y Automática, Escuela Superior de Ingeniería Bilbao. Open Course Ware.[En línea] disponible en http://ocw. ehu. es/ensenanzas-tecnicas/redes-neuronales-artificiales-y-sus-aplicaciones/Course_ listing.[Consultada 20-09-2012]*, 2008.

[7] M. Caballero, S. Lozano, and B. Ortega. Efecto invernadero, calentamiento global y cambio climático: una perspectiva desde las ciencias de la tierra. *Revista digital universitaria*, 8, 2007.

[8] J. Cohen, C. F. Crispim-Junior, C. Grange-Faivre, and L. Tougne. CAD-based Learning for Egocentric Object Detection in Industrial Context.

In *15th International Conference on Computer Vision Theory and Applications*, volume 5, Valletta, Malta, Feb. 2020. SCITEPRESS - Science and Technology Publications.

[9] G. Cortina Fernández. Técnicas inteligentes para su integración en un vehículo autómata. Trabajo de Fin de Grado en Ingeniería del Software, Facultad de Informática UCM, Departamento de Ingeniería del Software e Inteligencia Artificial, Curso 2019/2020., 2020.

[10] B. Cyganek. *Object Detection and Recognition in Digital Images: Theory and Practice*. Wiley, 2013.

[11] R. Flórez López, J. M. Fernández, and J. M. Fernández Fernández. *Las Redes Neuronales Artificiales*. Metodología y Análisis de Datos en Ciencias Sociales. Netbiblo, 2008.

[12] R. Fonfría, R. Sans, and J. de Pablo Ribas. *Ingeniería ambiental: contaminación y tratamientos*. Colección productiva. Marcombo, 1989.

[13] S. Frintrop. *VOCUS: A visual attention system for object detection and goal-directed search*, volume 3899. Springer, 2006.

[14] L. García Rodríguez. *Algunas cuestiones notables sobre el modelo de Hopfield en optimización*. PhD thesis, Madrid, Noviembre 2018. Tesis de la Universidad Complutense de Madrid, Facultad de Ciencias Matemáticas, Departamento de Estadística e Investigación Operativa, leída el 15-12-2017.

[15] G. A. Gómez Rojas, J. C. Henao López, and H. Salazar Isaza. Entrenamiento de una red neuronal artificial usando el algoritmo simulated annealing. *Scientia Et Technica*, 2004.

[16] G. Guridi Mateos et al. Modelos de redes neuronales recurrentes en clasificación de patentes. B.S. thesis, 2017.

[17] J. R. Hilera and V. J. Martínez Hernando. *Redes neuronales artificiales: fundamentos, modelos y aplicaciones*. 01 1995.

[18] S. Hinterstoisser, S. Benhimane, V. Lepetit, P. Fua, and N. Navab. Simultaneous recognition and homography extraction of local patches with a simple linear classifier. In *Proceedings of the British Machine Vision Conference*. BMVA Press, 2008. doi:10.5244/C.22.10.

[19] J. J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558, 1982.

[20] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications, 2017.

[21] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <0.5mb model size, 2016.

[22] A. Iguarán Guerra, S. Gómez Ruíz, et al. Análisis de las necesidades y dificultades en la disposición de residuos sólidos en la fuente doméstica para el desarrollo de un producto. B.S. thesis, Universidad EAFIT, 2010.

[23] Y.-C. Jhang, A. Palmar, B. Li, S. Dhakad, S. K. Vishwakarma, J. Hogins, A. Crespi, C. Kerr, S. Chockalingam, C. Romero, A. Thaman, and S. Ganguly. Training a performant object detection ML model on synthetic data using Unity Perception tools, Sep 2020.

[24] R. Karim. *TensorFlow: Powerful Predictive Analytics with TensorFlow*. Packt Publishing, Limited, 2018.

[25] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[26] P. Larranaga, I. Inza, and A. Moujahid. Tema 8. redes neuronales. *Redes Neuronales, U. del P. Vasco*, 12, 1997.

[27] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár. Microsoft coco: Common objects in context, 2015.

[28] P. López and J. García-Consuegra Bleda. *Informática gráfica*, volume 19. Univ de Castilla La Mancha, 1999.

[29] M. A. López Pacheco. Identificación de sistemas no lineales con redes neuronales convolucionales. *Cuidad de Mexico: Centro de investigación y de estudios avanzados*, 2017.

[30] D. J. Matich. Redes neuronales: Conceptos básicos y aplicaciones. *Universidad Tecnológica Nacional, México*, 41, 2001.

[31] W. S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4), 1943.

[32] A. J. McMichael, D. Campbell-Lendrum, S. Kovats, S. Edwards, P. Wilkinson, T. Wilson, R. Nicholls, S. Hales, F. Tanser, D. L. Sueur, M. Schlesinger, and N. Andronova. Chapter 20 global climate change.

[33] M. Minsky and S. A. Papert. *Perceptrons: An introduction to computational geometry*. MIT press, 2017.

[34] S. Morant Gálvez. Desarrollo de un sistema de bajo coste para el análisis de tráfico mediante el uso de deep learning. 2021.

[35] L. Moreno Díaz-Alejo. Análisis comparativo de arquitecturas de redes neuronales para la clasificación de imágenes. Master's thesis, 2020.

[36] B. Müller, J. Reinhardt, and M. Strickland. *Neural Networks: An Introduction*. Physics of Neural Networks. Springer Berlin Heidelberg, 1995.

[37] C. Parra Ramos and D. Regajo Rodríguez. Reconocimiento automático de matrículas. *Universidad Carlos III de Madrid*, 2006.

[38] R. Pavón Benítez. Técnicas de deep learning para el reconocimiento de movimientos corporales. Trabajo de Fin de Grado en Ingeniería del Software, Facultad de Informática UCM, Departamento de Ingeniería de Software e Inteligencia Artificial, Curso 2019/2020, 2020.

[39] A. Polacco and K. Backes. The amazon go concept: Implications, applications, and sustainability. *Journal of Business and Management*, 24(1), 2018.

[40] A. Polacco and K. Backes. The amazon go concept: Implications, applications, and sustainability. *Journal of Business and Management*, 24(1), 2018.

[41] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[42] F. Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[43] A. Rozantsev, V. Lepetit, and P. Fua. On rendering synthetic images for training an object detector. *Computer Vision and Image Understanding*, 137, 11 2014.

[44] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *nature*, 323(6088), 1986.

[45] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[46] R. Salas. Redes neuronales artificiales. *Universidad de Valparaíso. Departamento de Computación*, 1, 2004.

[47] M. Sánchez and J. Castro. *Gestión y Minimización de Residuos*. Fundación Confemetal, 2007.

[48] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[49] O. Simeone. A very brief introduction to machine learning with applications to communication systems. *IEEE Transactions on Cognitive Communications and Networking*, 4(4), 2018.

[50] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[51] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, June 2015.

[52] A. Terceño Ortega. Análisis de un modelo predictivo basado en google cloud y tensorflow. Trabajo de Fin de Grado en Ingeniería Informática y Matemáticas (Universidad Complutense, Facultad de Informática, curso 2016/2017), 2017.

[53] Unity Technologies. Unity Perception package, 2020.

[54] S.-C. Wang. *Artificial Neural Network*. Springer US, Boston, MA, 2003.

[55] P. Warden and D. Situnayake. *TinyML: Machine Learning with TensorFlow Lite on Arduino and Ultra-Low-Power Microcontrollers*. O'Reilly Media, 2019.

[56] W. Yu and Y. Bai. Visualizing and comparing alexnet and vgg using deconvolutional layers. 2016.

[57] J. Zamorano Ruiz et al. Comparación y análisis de métodos de clasificación con las bibliotecas scikit-learn y tensorflow en python. 2019.

[58] J. Zurada. *Introduction to Artificial Neural Systems*. West, 1992.