



# RECONOCIMIENTO DE IMÁGENES CON REDES CONVOLUCIONALES EN MATLAB

ARTÍCULO CIENTÍFICO

RECIBIDO: 15/07/2019 ACEPTADO: 15/09/2019 ÚLTIMA VERSIÓN: 15/02/2020

## *AN APPROACH TO THE PROCESSING OF IMAGES AND IMPLEMENTATION WITH SOFTWARE IN THE INDUSTRY*

Joseph Jhonas Vogulys Medina

DAVINCHI, INGENIERÍA MECATRÓNICA., Univ. Militar Nueva Granada. Email: u3900234@unimilitar.edu.co

Como citar este artículo: Vogulys, J, (2020) Reconocimiento de imágenes con redes convolucionales en MATLAB, revista científica Dynamikos 1 (1). Recuperado de: <https://teinco.edu.co/revistas-2-2/>

### RESUMEN

Las técnicas de visión artificial han tenido un gran avance en el mundo gracias a la fiabilidad de su respuesta reconociendo patrones superando en algunos casos la visión humana. Estas técnicas se han utilizado de reconocimiento de números, personas, rostros, y hasta en la conducción autónoma. En este caso se pretende abordar la utilización de un método de aprendizaje profundo con el uso del software Matlab, el cual con un pequeño algoritmo y una base de datos relativamente pequeña de imágenes (algunas centenas), se logre entrenar una red neuronal convoluciones que en este caso particular serán dos categorías, semáforos en verde y en rojo, evidenciando una metodología sencilla la cual puede ser utilizada para el reconocimiento de casi cualquier imagen con una grado de asertividad de casi el 90%.

**PALABRAS CLAVE:** Redes neuronales Convolucionales, Aprendizaje Profundo, visión artificial.

### ABSTRACT

Artificial vision techniques have made great progress in the world thanks to the reliability of their response recognizing patterns in some cases surpassing human vision. These techniques are used to recognize numbers, people, faces, and even in autonomous driving. In this case it is intended to address the use of a deep learning method with the use of Matlab software, which with a small algorithm and a relatively small database of images (a few hundred), will be able to train a neural network convolution that in This particular case will be two categories, traffic lights in green and red, evidencing a simple methodology which can be used for the recognition of almost any image with an assertiveness degree of almost 90%.

**KEYWORDS:** Convolutionary neural networks, Deep Learning, artificial vision.

## INTRODUCCIÓN

Recientemente el Aprendizaje profundo (DL) (Bengio, Y. 2009), ha mostrado un gran Éxito en el aprendizaje supervisado en diversas áreas del procesamiento de imágenes, el procesamiento de la información con cámaras se ha estudiado en hogares (Esteve A 2019), se presenta el diseño de un sistema de detección de peatones usando redes neuronales convoluciones (CNN) enfocado en vehículos autónomos.

Aunque la AI, (Wei, W., He, S., Wang, D., 2018) (Inteligencia Artificial) tiene diversas ramas de trabajo el DL (Deep Learning o aprendizaje profundo) ha tenido mejores resultados frente al ML (Maching Learning) como se evidencia en la figura 1.

### Why is Deep Learning so popular now?

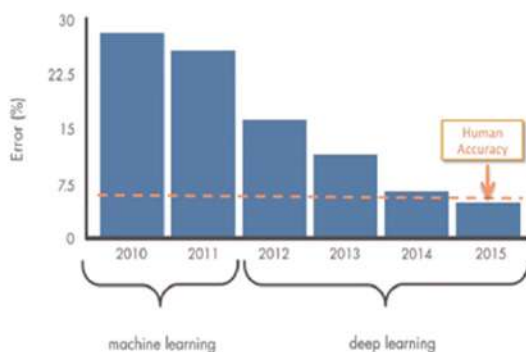


Figura 1: Error vs precisión del ML y el DL. Fuente: Mathworks

El ML utiliza algoritmos conocidos de clasificación como árboles de decisión, máquina de vector soporte, KNN vecino más próximo, regresión logística entre otros, utilizando etiquetas no obtuvieron el alcance que los algoritmos del aprendizaje profundo (Y. LeCun, 2015). En (Romanov, N., Johnson, C. E2015), se evidencia la efectividad del DL en campos de la salud y análisis microscópicos. Como conclusión se puede decir que el error cuadrático medio utilizado para comparar el accuracy o asertividad presenta mejores resultados el Deep Learning que el Maching Learning.

Recientemente, Aprendizaje profundo ha mostrado una gran Éxito en el aprendizaje supervisado en diversas

áreas como el reconocimiento de imágenes (J. Deng, W. Dong, R. Socher 2009), reconocimiento de voz Zeiler, (M. D., and Fergus, R. 2014), análisis de sentimiento, juegos y navegación de robot. En la práctica, el aprendizaje automático con imágenes como entradas se logra mediante Redes neuronales convolucionales (CNN), que se modelan imitando el procesamiento de la corteza visual de los mamíferos, (Fan, Q., Brown, L., & Smith, J 2016), entre ellas redes conocidas como AlexNet (S. J.Mankar 2017) y GoogLeNet, Los modelos CNN se han convertido una alternativa excepcional gracias a su desempeño en reconocimiento imagen.

Los primeros avances obtenidos en el reconocimiento de imágenes se dan en 2012 evaluando por barrido de píxeles con números escritos a mano (D. Silver, A. Huang 2016), gracias al concurso ImageNet el cual convoca académicos en el mundo para probar la asertividad y velocidad de identificación, obteniendo como ganador una red neuronal convoluciones usando 1000 categorías, es decir la red es capaz de reconocer mil objetos entre ellos animales, objetos, utensilios etc. Hoy consta de cientos de miles de imágenes completamente segmentadas con ImageNet, que consiste en Más de 15 millones de imágenes etiquetadas de alta resolución en más de 22,000 categorías que sirven para aprender sobre miles de objetos de millones de imágenes, y gracias a ellas no necesitaremos sino algunas centenas para entrenas nuestra categoría de imágenes.

La detección de objetos ayuda en la estimación de posturas, detección de vehículos, vigilancia, etc.

## METODOLOGÍA

### Entender las Redes convoluciones

En Deep Learning el proceso difiere a los que conocemos como una red neuronal multicapa convencional como las famosas Adaline utilizando una gran serie de conexiones. Una red neuronal se conforma por pesos sinápticos, neuronas y capas que se conforman por variables de entrada, capas ocultas y de salida.

Dentro de cada neurona podemos decir que es una regresión lineal donde los pesos le dan la inclinación a la pendiente y el bias la traslación de esta, y se parametriza según la función de activación sea escalón, rampa, o sigmoide y posteriormente se busca el mínimo global por medio del famoso gradiente descendiente que no es más que un cálculo de derivadas parciales sin embargo se utiliza el conocido Back Propagación para reducir el consumo de máquina. En el cálculo del backpropagation se utiliza lo que conocemos como Error cuadrático medio, la función de activación, el bias.

En las redes convolucionales difiere el ehcho de que se trabaja por píxeles pasando de ser redes neuronales a Redes Convolucionales CNN [25] que emplean múltiples capas convolucionales, capas de normalización, pooling, redes neuronales totalmente conectadas y capas de clasificación.

En la figura 2.1 extraída de Mathwork se muestra la arquitectura de una CNN analizando una sección de la imagen de un coche, siendo esta una entrada, luego realiza una convolucion Relu, Pooling reiterativamente hasta llegar en forma de matrices para un análisis multidimensional a una capa fully conectada donde se analiza los patrones y se clasifican junto al Softmax.

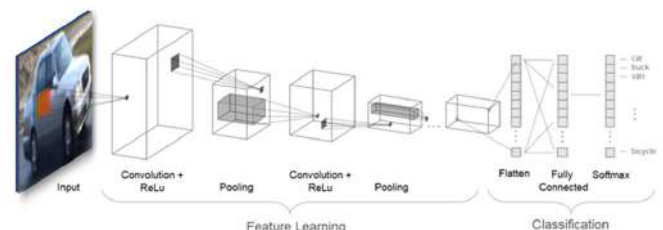


Figura 20-2: Arquitectura de una CNN. Fuente: Mathworks

En la convolución y en las capas ocultas de la red, se presentan los hiperparámetros, que se entienden como aquellos parámetros que ayudan a controlar el comportamiento del algoritmo de aprendizaje profundo y están especificados por el desarrollador. En la figura 2.2, se muestran los hiperparámetros que normalmente se encuentran en una capa oculta de una CNN, donde H es la altura y el ancho W de la imagen, F el tamaño o dimensión del filtro donde puede ser diferente tanto para el ancho como para la altura, S es el Paso o stride, P el relleno o padding Y D depth o la profundidad.

### Convolution Layer – Choosing Hyperparameters

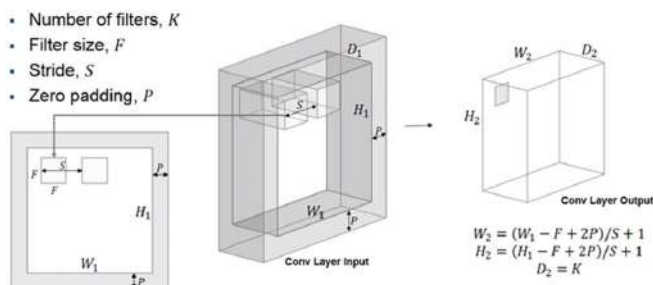


Figura 20-2: Hiperparámetros de la operación de las capas de la CNN.  
Fuente: Mathworks

### Dimensiones del filtro (F):

La dimensión del filtro o Kernel está representada por F, esto indica el tamaño de la disposición con la que se realizará la operación en cada capa oculta. Su forma puede ser rectangular o cuadrada y su tamaño puede variar en cada capa de la red dependiendo de las dimensiones características que desee resaltar. Además, puede definir la cantidad de filtros diferentes que desea aplicar a una imagen para obtener una mayor cantidad de funciones.

### Stride:

Representa el salto en píxeles tanto en H como en W del filtro que se aplica, siendo este valor uno de los más representativos para reducir el volumen de salida de la matriz o imagen de entrada. La elección correcta de este valor puede permitir que las capas de Agrupación o Pooling reduzcan la imagen mientras mantienen las características principales de la entrada original, el valor de este parámetro debe ser menor o igual al tamaño del filtro para evitar pérdidas en el aprendizaje de las características.

### Padding:

Hay casos en que algunas de las características que desea aprender la red están en los bordes de la imagen de entrada, cuando el filtro se aplica en una operación de convolución, no tiende a aprender las características del borde de la imagen. Para resolver este problema, se define una dimensión de relleno, es decir, se agregan filas y columnas en los bordes de la imagen para que el filtro aplicado pueda moverse más allá de la altura y el ancho de la imagen.

### Profundidad:

Son los canales de la disposición de entrada, una imagen en grises tiene un canal, un RGB tiene tres canales, mientras que un RGB-D tiene 4 canales.

Según sean los hiperparámetros seleccionados, el volumen de salida de cada capa cambia, esta salida es definida por la ecuación 1 y 2, presentadas en [25].

$$W = \frac{(W - F_w + 2P)}{S} + 1$$

$$H = \frac{(H - F_H + 2P)}{S} + 1$$

Cada una de las capas de la CNN, tiene una tarea específica que causan que al diseñar una arquitectura realizando una combinación de esta, la red sea capaz de aprender las características que causaran que pueda clasificar las imágenes en las categorías correctas.

### Convolución:

Las capas de las CNN combinan parámetros que permiten el aprendizaje de patrones para categorizarlas correctamente

$$s(t) = \int x(a)w(t-a)da \quad (3)$$

Esta se da al multiplicar dos funciones una se encarga de desplazar mientras que la Convolución es una sumatoria entre el filtro y el arreglo siendo I las entradas en píxeles y K el filtro.

$$S(i, j) = \sum_m \sum_n I(m, n)K(i - m, j - n) \quad (4)$$

Los filtros extraen patrones para que la red aprenda como se verá a continuación bidimensionalmente figura 2.3

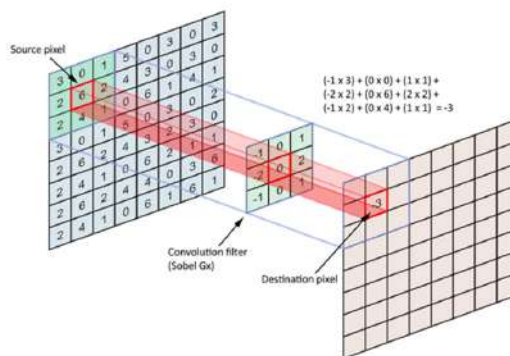


Figura 2-3: Operación de Convolución.

### Rectified Linear Unit (ReLU):

Unidad de rectificación lineal o RELU, no es más que una función de activación que solo obtiene los números positivos después de la convoluciones Romanov, (N., Johnson, C. E 2016). En la figura 2.4 se puede observar el rango de operación de la una RELU.

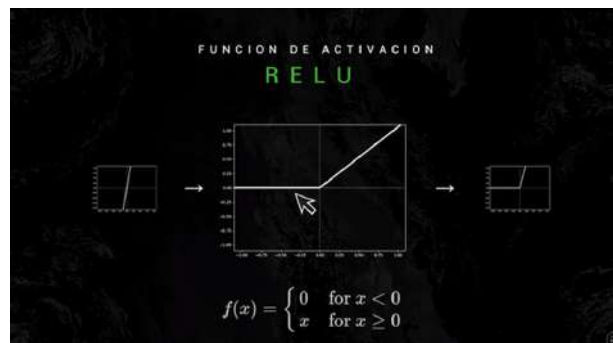


Figure 20-4: Rectified Linear Unit (ReLU). (N.,Johnson, C. E 2016)

**Pooling:**

Agrupar caracteres es importante para reducir información dependiendo si es maxpooling o average pooling, se toma un número de cada arreglo de píxeles pasando una imagen a ser por ejemplo 4 veces más pequeña ver figura 2-5.

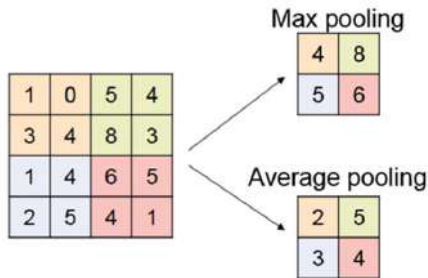


Figura 20-5: Dropout en una red neuronal

**Normalización:**

En la normalización se utilizan dos técnicas conocidas como la Batch y la Group Normalization siendo C las filas y N las columnas y así evitar confundir la red Batch Normalization y el Group Normalization Ioffe, S., and Szegedy, C. (2015).

Las redes neuronales convolucionales trabajan modelando de forma consecutiva pequeñas piezas de información, y luego combinando esta información en las capas más profundas de la red. Una manera de entenderlas es que la primera capa intentará detectar los bordes y establecer patrones de detección de bordes. Luego, las capas posteriores tratarán de combinarlos en formas más simples y, finalmente, en patrones de las diferentes posiciones de los objetos, iluminación, escalas, etc. Las capas finales intentarán hacer coincidir una imagen de entrada con todos los patrones y arribar a una predicción final como una suma ponderada de todos ellos. De esta forma las redes neuronales convolucionales son capaces de modelar complejas variaciones y comportamientos dando predicciones bastante precisas.

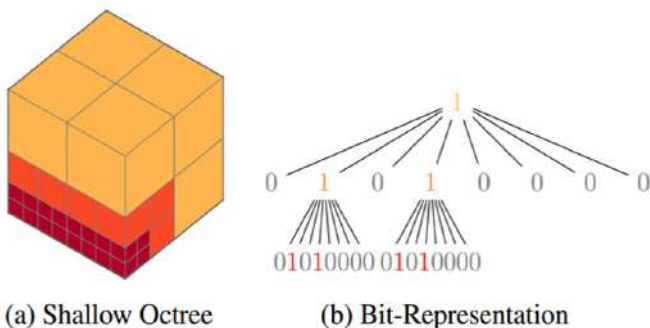


Figure 20-6: Batch Normalization y Group Normalization

**Fully Connected:**

La etapa fully connected es la red neuronal multicapa convencional y esta al final de la arquitectura ya que toma las características más importantes.

**Dropout:**

En el aprendizaje puede ocurrir un over fitting o sobre ajuste el cual genera problemas cuando se entrena una red ya que aprende tanto unos parámetros que cuando se pone a prueba es incapaz de reconocerlo ya que prácticamente ha memorizado las entradas de entrenamiento, por eso el dropout sirve desconectando aleatoriamente neuronas disminuyendo el sobre ajuste para que reconozca ahora si entradas similares sin problema (Bengio, Y.2017), en la figura 2-7 se puede observar la arquitectura de una red neuronal multicapa.

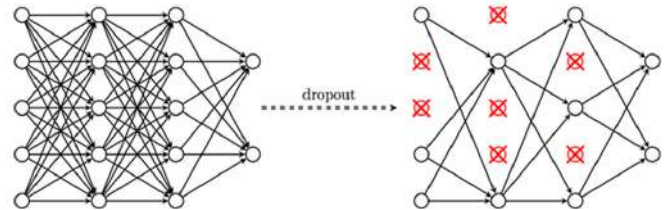


Figura 20-7: Dropout en una red neuronal (Bengio, Y.2017)

**Softmax:**

El softmax es aquel que da un valor de 0 a 100% en semejanza de las imágenes aprendidas. Por ejemplo, al probar con la imagen de un gato, seguramente obtendría un 90% de probabilidad de ser gato, sin embargo, también podría pensar que es en un porcentaje parecido a algún felino e incluso a algún otro animal similar, obviamente la probabilidad será cada vez menor. (M. Y. Day and Y. D. Lin, 2017).

**2.2 Proceso de entrenamiento de la red****2.2.1 Recolectar y etiquetar imágenes extraídas en las calles de Bogotá, para el entrenamiento de la red convolucional.**

En esta etapa se pretende recolectar imágenes de semáforos con fotografías y de internet.

**Subtareas:**

1. Se deberán guardar en formato [227 x 227] Píxeles para poder ser leída por AlexNet y [224 x 224] píxeles para ser leída por GoogleNet.
2. Usar bases de datos con imágenes cuadradas que no deformen sus características y guardarla en una carpeta llamada "SEMAFOROS". Dentro de esta en una carpeta con el nombre de "VERDES" (aquí se agregan al menos 80 imágenes de semáforo verdes) y otra carpeta con el nombre de "ROJOS" (aquí se agregan al menos 80 imágenes de semáforos en rojo)

**2.2.2 Entrenar las redes neuronales convoluciones.**

Aquí se pone a correr el código para entrenar la red neuronal.

**Subtareas:**

1. Se deberá agregar un código en el script de Matlab Se debe ejecutar con la ruta donde se observe la carpeta "SEMAFOROS".
2. El código le solicitará al usuario descargar la red convolucional AlexNet o GoogLeNet de Matlab para poder ser ejecutada correctamente.

### 2.2.3 Probar el correcto funcionamiento del código.

En esta etapa se pretende encontrar un modelo óptimo para entrenar comparándolo con la matriz de confusión y el error vs el accuracy.

#### Subtareas:

Se Deberá evidenciar que el Accuracy sea mayor al 85% Se probará con video su efectividad.

Crear código para ver que funciona en video

#### Código a utilizado

```
%Se importa la CNN de alexnet, se carga la base de datos, 'semaforos';
%Se donde se ubica la base de datos; includesub es tener en cuenta las
%carpetas, los label son las etiquetas, foldernames, readfcn bautiza como funcion,
%semread

lemon_ds = imageDatastore('semaforos','IncludeSubfolders',true,...
    'LabelSource','foldernames',...
    'ReadFcn',@semread);

%lemon, 0,9 es la cantidad de imagen a entrenar, es decir el 0.1 para probar
%aleatoriamente, randomized
%numclases, numerar cantidad
[trainings,testings] = splitEachLabel(lemon_ds,0.9,'randomized');
numclases = numel(categories(lemon_ds.labels));
net = alexnet; %carga alexnet
layers = net.layers; %se extraen 25
%entrenar el numclases es decir no 1000,sino 2 categorias, que es la capa 23
layers(end-2) = fullyConnectedLayer(numclases);

layers(end) = classificationLayer; %cambia de mil etiquetas a 2.
miniBatchSize = 1; % La dimension de imagenes.

% Al aumentar las epocas mejora el aprendizaje
maxEpochs = 10;
%factor de aprendizaje, debe ser pequeno para hacer pasos mas pequenos. o lane rate
lr = 0.00001;
%gradiente descendiente con momento w1+alfa*De/Dw+momento*dw(k-1)
%el momento mira la inclinacion buscando la caida segun el factor de aprendizaje
opts = trainingOptions('sgdm', ...
    'InitialLearnRate', lr,...
    'LearnRateSchedule','none',...
    'MaxEpochs', maxEpochs, ... %
    'MiniBatchSize', miniBatchSize,...
    'VerboseFrequency',1); %visualizacion del entrenamiento de los batch

%trainings=augmentedImageSource([227 227],trainings)
%En la variable info podremos observar el Traininloss y TrainingAccuracy:

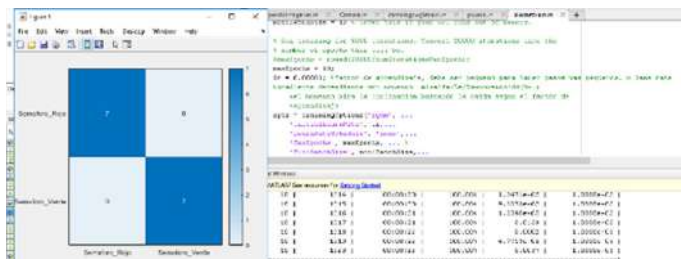
[lemonnet,info] = trainNetwork(trainings, layers, opts);

%Si deseamos podemos plotear estos datos:
plot(info.TrainingLoss)
lemPreds=classify(lemonnet,testings) %veo el test vs el 90% entrenado
lemActual=testings.Labels %
numCorrect=nnz(lemPreds==lemActual)
fracCorrect=numCorrect/numel(lemActual) %precision de la red
[lemconf,lemnames]=confusionmat(testings.Labels,lemPreds)
heatmap(lemnames,lemnames,lemconf)

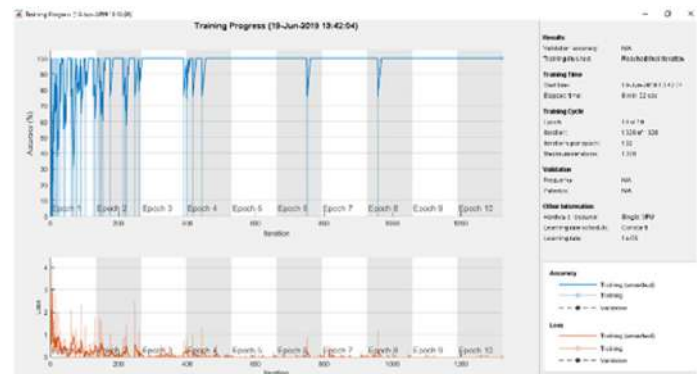
function img = semread(file)
    img = imread(file); %importa la imagen
    img = imresize(img,[227 227]); %Redimensiona a 227 pixeles
end
```

### 3. Resultados

Se evidencia gracias a la matriz de confusión que el 10% de las imágenes testeadas fueron reconocidas perfectamente es decir 7 rojas y 7 verdes.



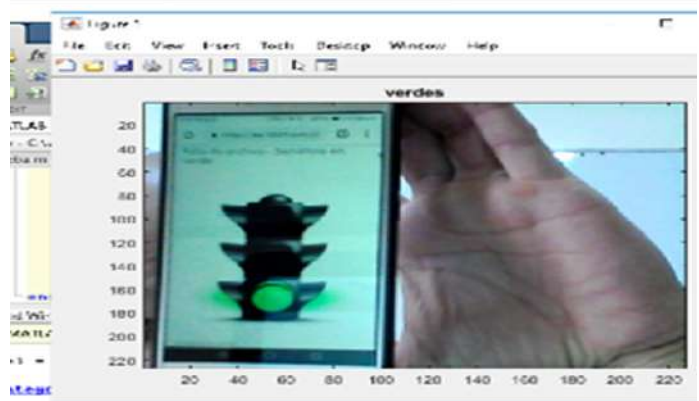
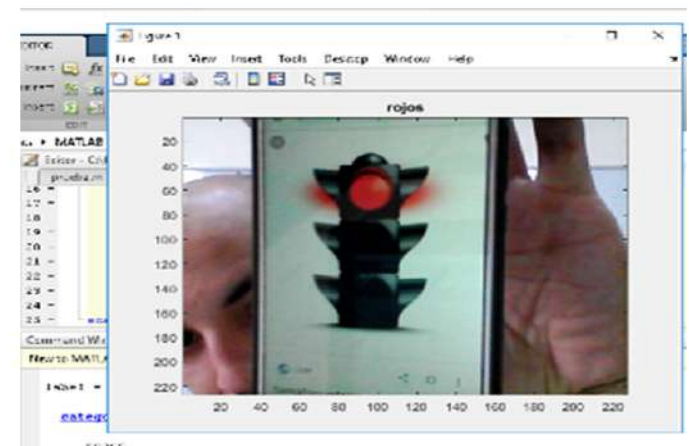
Se obtiene un gráfico donde se evalúa el error vs las iteraciones



Aquí se evidencia que el Accuracy (%) desde las 3ra época se empieza a estabilizar en casi 100% para desde la 8 a la 10ma época está prácticamente entrenada.

El tiempo de entrenamiento fue de 8 minutos 22 segundos, hubo 1320 iteraciones.

El error o loss disminuye a cero con pocas oscilaciones a partir de la 500 iteración aproximadamente.



Se evidencia que el semáforo verde se reconoce con un 90.17% y el rojo con un 88.58%

Luego se comparó con el app de classification learner de Matlab y se obtiene la siguiente matriz de confusión, con regresión logística en un 80% de confiabilidad



A lo que se evidencia que la técnica de Deep learning con redes convolucionales es más eficiente con un accuracy que tiende al 100%

## CONCLUSIONES

Se evidencia que el Deep learning refleja mejores resultados que las técnicas clásicas de machine learning.

Se observa también que con pocas líneas de código se pueden obtener muy buenos resultados gracias a los avances en las redes convolucionales, apalancándose con redes creadas como AlexNet, GoogleNet, ResNet 18 entre otros, muy buenos resultados utilizando un número pequeño de imágenes y se pueden comparar gracias a la matriz de confusión y el gráfico de Loss vs iteraciones que Matlab provee.

En conclusión, gracias a Matlab se pueden lograr técnicas similares a las obtenidas con el uso de Tensor Flow o Keras, pero este facilita el trabajo gracias a su lenguaje de alto nivel.

Tabla 1 Abreviaciones Sugeridas para la Lista de Referencias

| Relación de abreviaciones Permitidas                     |                 |
|----------------------------------------------------------|-----------------|
| Palabra                                                  | Forma Abreviada |
| AI = Inteligencia Artificial, (Artificial Intelligence). | AI              |
| Redes Neuronales Convolucionales                         | CNN             |

| Relación de abreviaciones Permitidas      |                 |
|-------------------------------------------|-----------------|
| Palabra                                   | Forma Abreviada |
| Aprendizaje Profundo (Deep Learning)      | DL              |
| Aprendizaje de Máquina (Machine Learning) | ML              |



## REFERENCIAS

- A. Esteva, B. Kuprel, R. A. Novoa, J. Ko, S. M. Swetter, H. M. Blau, and S. Thrun, (2017) "Dermatologist-level classification of skin cancer with deep neural networks," *Nature*, vol. 542, no. 7639, pp. 115–118,
- A. Krizhevsky, I. Sutskever, & G. E. Hinton, (2012). ImageNet Classification with Deep Convolutional Neural Networks. *Advances in Neural Information Processing Systems 25 (NIPS 2012)*. pp. 1106– 1114.
- Bengio, Y. (2009). Learning deep architectures for AI. *Foundations and trends® in Machine Learning*, 2(1), 1-127.
- Bengio, Y., Ducharme, R., Vincent, P., and Jauvin, C. (2003). A neural probabilistic language model. *Journal of machine learning research*, 3, pp. 1137-1155.
- C. Spille, S.D. Ewert, B. Kollmeier, B. Meyer, (2018) Predicting speech intelligibility with deep neural networks, *Computer Speech & Language* 4851-66.
- C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, & A. Rabinovich. (2015). Going deeper with convolutions. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, 2015, pp. 19.

- D. Silver, A. Huang, C. Maddison, A. Guez, L. Sifre et al., (2016) “Mastering the game of go with deep neural networks and tree search,” *Nature*, vol.529, no. 7587, pp. 484–489, 2016.
- Esteva, A., Kuprel, B., Novoa, R. A., Ko, J., Swetter, S. M., Blau, H. M., & Thrun, S. (2017). Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, 542(7639), 115. with deep neural networks,” *Nature*, vol. 542, no. 7639, pp. 115–118, 2017.
- Fan, Q., Brown, L., & Smith, J. (2016) June. A closer look at Faster R-CNN for vehicle detection. In 2016 IEEE intelligent vehicles symposium (IV) (pp. 124-129). IEEE.
- G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, & R. R. Salakhutdinov, (2012). Improving neural networks by preventing coadaptation
- Goodfellow, I., Bengio, Y., Courville, A., and Bengio, Y. (2016). *Deep learning* (Vol. 1). Cambridge: MIT press.
- Ioffe, S., and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*.
- J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. (2009) ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.
- K. M. I. Khalilullah, S. Ota, T. Yasuda and M. Jindai, (2017) “Development of robot navigation method based on single camera vision using deep learning,” 2017 56th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE), Kanazawa, Japan, 2017, pp. 939-942, doi: 10.23919/SICE.2017.8105675
- Krizhevsky, A., Sutskever, I. and Hinton, G.E., (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (pp. 1097-1105).
- LeCun, Y., Bengio, Y., and Hinton, G. (2015). *Deep learning*. *nature*, 521(7553), p. 436.
- LeCun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W. and Jackel, L.D., (1989). Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4), pp.541-551.
- M. Y. Day and Y. D. Lin, (2017) “Deep Learning for Sentiment Analysis on Google Play Consumer Review,” 2017 IEEE International Conference on Information Reuse and Integration (IRI), San Diego, CA, USA, 2017, pp. 382-388, doi: 10.1109/IRI.2017.
- Romanov, N., Johnson, C. E., Case, J. P., Goel, D., Gutmann, S., & Dooley, M. (2015). U.S. Patent No. 8,961,695. Washington, DC: U.S. Patent and Trademark Office.
- Russell, S. J., & Norvig, P. (2016). *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited, Schmidhuber, J. (2015). *Deep learning in neural networks: An overview*. *Neural networks*, 61, 85-117.
- S. J. Mankar, et al., (2016) “Design of computer vision intelligent system for lane detection,” in *Green Engineering and Technologies (IC-GET)*, Coimbatore, India, 2016.
- Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1), pp. 1929-1958.
- Wei, W., He, S., Wang, D., & Yeboah, Y. (2018, June). Multi-Objective Deep CNN for Outdoor Auto-Navigation. In *Proceedings of the 2018 2nd International Conference on Deep Learning Technologies* (pp. 81-85). ACM.
- Wu, Y., and He, K. (2018). Group normalization. *arXiv preprint arXiv:1803.08494*.
- X. Zhang et al., (2017) “Classification of Mammographic masses by deep learning,” 2017 56th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE), Kanazawa, Japan, 2017, pp. 793-796, doi: 10.23919/SICE.2017.8105545205
- Yang, S., Wang, J., Wang, G., Hu, X., Zhou, M. and Liao, Q., (2017), December. Robust RGB-D SLAM in dynamic environment using faster R-CNN. In *Computer and Communications (ICCC)*, 2017 3rd IEEE International Conference on (pp. 2398-2402). IEEE.
- Y. LeCun, Y. Bengio, and G. Hinton. (2015) “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, 2015. of feature detectors. (<https://arxiv.org/pdf/1207.0580.pdf> accessed on Aug. 20, 2017).
- Y. Jia, E. Shelhamer, J. Donahue, et al., (2014) “Caffe: convolutional architecture for fast feature embedding,” *arXiv preprint arXiv:1408.5093*, 2014.
- Zeiler, M. D., and Fergus, R. (2014, September). Visualizing and understanding convolutional networks. In *European conference on computer vision* (pp. 818-833). Springer, Cham.